UCRL-MA-121327 Rev. 1

CombinePlt and CombineThs User Manual

Merging Multiple, Processor-Local Plot and Time-History Data Bases Produced During a Parallel Calculation*

Richard J. Procassini
Anthony J. DeGroot

September 21, 1995

Lawrence Livermore National Laboratory

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

DISCLAIMER

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial products, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

CombinePlt and CombineThs User Manual

Merging Multiple, Processor-Local Plot and Time-History Data Bases Produced During a Parallel Calculation *

Richard J. Procassini

Anthony J. DeGroot

Lawrence Livermore National Laboratory

Livermore, CA 94551

21 September 1995

Abstract

The CombinePlt and CombineThs post-processing utilities are designed to merge the data in multiple, processor-local plot and time-history data bases produced by the parallel versions of the analysis codes DYNA3D [1], NIKE3D [2] or PING [3] into a serial data base which is compatible with the existing versions of the GRIZ [4] and THUG [5] visualization tools. These utilities make use of the partition assignment file produced by the PartMesh [6] suite of pre-processing utilities to map the data from the processor-local order to global order. These utilities are also capable of translating 64-bit IEEE data bases into 32-bit IEEE data bases which are required for post-processing with GRIZ or THUG on an SGI workstation.

*Work performed under the auspices of the U.S. Department of Energy by the Lawrence Livermore National Laboratory under contract No. W-7405-ENG-48

1 Introduction

The lack of a parallel I/O library which (*i*) allows multiple processors to read from or write to a single "logical" file and (*ii*) runs efficiently on a large number of parallel computing platforms has motivated the implementation of the current I/O scheme within the parallel versions of the analysis codes DYNA3D [1], NIKE3D [2] and PING [3]. With regard to the output files produced by these codes, this I/O scheme writes a separate set of data files of a given type (i.e. plot, time-history and restart data bases) from each processor that takes part in the calculation. Each processor writes *only* the data that is associated with its subdomain, according to the spatial domain decomposition information in the partition assignment file which is produced by the PartMesh [6] pre-processing utility.

The difficulty with this I/O scheme is evident when one realizes that the visualization post-processing tools GRIZ [4] and THUG [5] currently operate only in a serial mode. Recent experience has shown that the analysis codes are usually the first to be ported to run on parallel computing platforms, followed at a later data by the pre- and post-processing tools. While it is possible to analyze the data in each of the processor-local data bases independently, this approach becomes unmanageable as the number of processors increases into the massively parallel realm of computing. It is also not clear that analysis of individual subdomains will provide the same degree of insight that the analysis of the complete domain provides. This difficulty in analysis is further complicated by domain decompositions of unstructured meshes which do not produce spatially contiguous subdomains.

As an interim solution to the problem of parallel I/O and visualization of unstructured mesh problems, we have developed a set of post-processing utilities that merge the multiple data bases produced by a parallel calculation into a single data base with the data in global order. The CombinePlt and CombineThs merging utilities are designed to perform this function for the plot and time-history data bases, respectively. These are serial utilities that can be executed either on a workstation or on a single node of a parallel computer. The only requirement of the computing platform is that it allow for a 32-bit IEEE word format, since this is the format required by GRIZ and THUG to analyze the data files on an SGI workstation. The multiple sets of data files produced by the individual processors in a parallel calculation are processed in a sequential fashion. The utilities read in the processor-local data from each of the input data bases and then map the data into global order prior to writing it to a single output data base.

The following section discusses how to execute the CombinePlt and CombineThs utilities, as well as how the various sections of the plot and time-history data bases are organized within the input and output data base files. The layout of data within the current (TAURUS compatible) version of the plot and time-history data bases is defined in Sections 3 and 4, respectively.

2 The CombinePlt and CombineThs Utilities

Once the analysis codes have produced N_P independent sets of plot or time-history data base files, where N_P is the number of processors taking part in the calculation, the CombinePlt and CombineThs utilities are used to merge the data sets into a single plot or time-history data base. The procedure maps the data in processor-local order in the input data files into global order in the output data file.

2.1 Command Line Arguments for the Merging Utilities

To execute either of the data-base merging utilities, type:

`combinedb` *input_root_name* *output_root_name* *partition_name* *input_size_flag* *material_flag*

where the suffix "`db`" is either "`plt`" for a plot data base or "`ths`" for a time-history data base. The arguments *input_root_name* and *output_root_name* are required to specify which data base files are to be processed and what the resultant data base files are to be called. The name of the partition assignment file produced by the PartMesh pre-processing utility is to be supplied for the argument *partition_name*. The argument *input_size_flag* specifies the word size of the files that are to be processed: 0 for 32-bit IEEE-format files or 1 for 64-bit IEEE-format files. Finally, the argument *material_flag* allows the user to specify the meaning of the material entry for each element in the data base: 0 for the material number in the input data base or 1 for the processor that the element is assigned to.

The parallel versions of the analysis codes currently produce familied output files which are named according to the following convention:

$$\texttt{input\_file\_name} = \textit{input\_root\_name}\mathbf{PID}\mathbf{nn}$$

where **PID** is the number of the processor which produced the file, in the range $0 \leq \mathbf{PID} \leq N_P - 1$, and nn is the family suffix number of the file.

The merging utilities produce familied files which are named as follows:

$$\texttt{output\_file\_name} = \textit{output\_root\_name}\mathrm{nn}$$

where nn is the family suffix number of the file.

The numbering convention for familied files is as follows: the first file in a family has no suffix, the next file has the suffix nn $= 01$, followed by nn $= 02$, etc. The length of the suffix will change to 3 characters for nn ≥ 100, to 4 characters for nn ≥ 1000, etc.

The format of the information in the partition assignment file *partition_name* is described in Reference [6]. The merging utilities read this file to determine:

(1) The number of processors, and hence the number of input data bases,

(2) The global number of nodal points and elements which comprise the mesh,

(3) The number of nodal points and elements assigned to each subdomain or processor, and

(4) The indices of the nodal points and elements assigned to each of the subdomains or processors.

The assignment information (item four) represents the local-to-global map which is used to rearrange the processor-local data in the input files into the global order in the output files.

The input word size argument *input_size_flag* allows the user to specify whether the analysis code which produced the data files was run on a 32-bit (Meiko CS-2, Intel Paragon, IBM SP-2) or 64-bit (Cray T3D) parallel computer. If the analysis codes wrote the data with a 64-bit IEEE word format (*input_size_flag* $= 1$), the merging utilities will convert the input data to 32-bit IEEE format prior to writing the output files. This permits analysis of data files produced by the Cray T3D MPP with the GRIZ and THUG post-processors on an SGI workstation.

The material description argument *material_flag* allows the user to specify the meaning of the material entry that is associated with each element in the geometry and connectivity section of the output (global) data base. For *material_flag* $= 0$ it represents the material number that is used to describe the constituitive properties of the element in the analysis code, while for *material_flag* $= 1$ it represents the number of the processor that the element was assigned to during the parallel calculation.

2.2 Data-Section Distribution Within and Across Files

Before proceeding to a detailed description of the data layout within the various sections of the the plot and time-history data bases (control data, geometry and connectivity data, and result data sections), it is useful to discuss how those sections are distributed within and across files. The situation is complicated by the fact that the data distribution scheme is different for the files produced by the analysis codes and by the merging utilities.

2.2.1 Files Produced by the Analysis Codes

The analysis codes employ an I/O package that was developed by Steven Sackett in the early 1980s which is based upon Fortran direct-access, binary reads and writes. This scheme utilizes a fixed-record length buffer for temporary data storage, which is filled prior to issuing a read or write command. While the length of this record/buffer L_{buf} is configurable at compile time, the convention has been to make this a power-of-two multiple of 512 words. By default, DYNA3D and NIKE3D use a 512 word record length, while PING uses a 4096 word record length.

The maximum size of any member of a familied set of files $L_{file,max}$ is also configurable at compile time. In addition, the size of the geometry/connectivity (L_{gc}) and result (L_{res}) data sections is problem dependent, as shown below in Sections 3 and 4. A given data base contains only one control data and one geometry/connectivity data section, but can contain several result data sections. This level of complexity has lead to the following convention for the distribution of data sections within and across files:

- If a file is large enough to contain an *integral* number of data sections, then it will do so.
- If a file is smaller than a given data section, then that data section will be *split* across multiple files.

- A data section will *only* be written to a file location other than the beginning of the file if the file is large enough to contain the entire data section.

In general, the individual files for a given data base may have a number of different lengths, depending upon the size of each of the data sections and the maximum file size. However, for each of these scenarios, the file size will always be an integral multiple of the buffer length ($L_{file} = M * L_{buf}$), *even if the amount of data that needs to be written to the file is* **not** *an integral multiple of the buffer length!* The "extraneous" information at the end of the files represents the remainder of a buffer which was previously written to the file.

Since the final file in a plot or time-history data base may be padded with extraneous information, the convention that is used to signal the end of the data base is to place a negative time word (usually -99999.0) directly after the last valid result data section in the data base.

2.2.2 Files Produced by the Merging Utilities

The merging utilities are written in C and employ a variable-record length, binary I/O scheme. This allows the merging utilities to utilize a much simpler data-section distribution scheme than the one discussed above:

- The control data and geometry/connectivity data sections are always written to the first familied file (the file without a suffix number).
- Each result data section is written to a separate file.

In general, this data-section distribution scheme results in a family of output files which have only two sizes, but with many more members comprising the family than the scheme described in the previous section.

The rearrangement of data sections within files that is inherent in the merging process also has a number of advantages relative to the previous scheme:

(1) There is no extraneous information at the end of the files.

(2) The file numbering scheme allows one to easily determine where to locate the result data section for a given simulation time. Such a determination is very complicated under the previous

scheme.

(3) The comparison of plot data bases produced on differing number of processors is relatively simple: each of the files in one output family can be `diffed` against those in the other family. Differing results can be localized to a given simulation time.

Note that the third item is only applicable to the plot data bases that are merged by CombinePlt. This type of comparison is possible because the result data for *each* of the nodal points and elements in the mesh are written in global order to the plot data base, which is a requirement for post-processing with GRIZ or TAURUS. In contrast, since (*i*) the time-history data base consists of result data for a *selected* subset of the nodal points and elements in the mesh, (*ii*) the nodal points and elements within a time-history block may be divided across processors (and even replicated across processors in the case of nodal points which lie on processor boundaries) and (*iii*) the THUG post-processor does not require that the time-history result data be ordered in a global sense, no attempt was made to write the result data in global order (or to remove replicated nodal point data) within the time-history data bases that are merged by CombineThs.

Since the data base files produced by the merging utilities do not contain extraneous information, there is no need to include an extra result-data section file with a negative time word to signal the end of the data base. The data base ends with the last sequentially number file in the family.

As a final note, it should be pointed out that the merging utilities may be used to rearrange the data in either a plot or time-history data base that is produced by a serial ($N_P = 1$) calculation. In this case, the user can manually create a simplified (three line) version of the partition assignment file which contains only the global nodal point and element information, as well as the number of processors (see Reference [6] for a complete description of the data format within the partition assignment file). When the number of processors is set to one, the merging utilities will not read the remainder of the partition assignment file, hence no local to global mapping is performed, and the net effect of the effort is to rearrange the data sections within the files.

3 Plot Data Base Description

The current form of the plot data base was originally designed to work with the TAURUS post-processor [7]. The format of this data base is also compatible for analysis with the GRIZ post-processor [4]. The plot data base is composed of at least three sections: a section of control data which provides the problem parameters, a geometry and connectivity section which defines the computational mesh, and at least one plot state or "snapshot" of various nodal and elemental results of the simulation for *all* of the nodal points and elements which form the computational mesh.

3.1 The Control Data Section

The control data section comprises the first 64 words of the plot data base. All of the entries in the control data section are integers. This section consists of problem description parameters in the order shown in Table 1.

The problem title consists of 10 words (40 characters for 32-bit word formats or 80 characters for 64-bit word formats). The flags IU, IV, IA and IT determine whether the nodal coordinates, velocities, accelerations and temperatures, respectively, are included for each plot state that is dumped to the data base.

Note that the variables for the number of materials used by beam, shell and hexahedral elements $NUMMAT2$, $NUMMAT4$ and $NUMMAT8$ are all ignored by GRIZ. The parameters $NV1D$, $NV2D$ and $NV3D$ represent the number of result variables for beam, shell and hexahedral elements that are written to the data base at each plot state:

- $NV1D = 6$ for DYNA3D/NIKE3D data bases ($ICODE = 2$ or $ICODE = 6$)
- $NV1D = 0$ for PING data bases ($ICODE = 300$)
- $NV2D = 33$ or $NV2D = 45$ for DYNA3D/NIKE3D data bases ($ICODE = 2$ or $ICODE = 6$)
- $NV2D = 0$ for PING data bases ($ICODE = 200$)
- $NV3D = 7$ for DYNA3D/NIKE3D data bases ($ICODE = 2$ or $ICODE = 6$)
- $NV3D = 0$ for PING data bases ($ICODE = 300$)

The activity data is included in the plot state section of the data base to specify whether each of the elements is included in the calculation at the time of the given plot state. If the variable $ACTIV$ is set in the range $1000 \leq ACTIV \leq 1005$, then element activity data is expected at each plot state

Table 1
The Control Data Section (Plot Data Base)

Variable	Words	Address	Description
TITLE	10	0	Problem Identification
Blank	5	10	
NDIM	1	15	Number of Dimensions: 2 - Two-Dimensional 3 - Three-Dimensional, Packed 4 - Three-Dimensional, Unpacked
NUMNP	1	16	Number of Nodal Points
ICODE	1	17	Code Descriptor: 2 - Pre-1988 DYNA3D or NIKE3D 6 - Post-1988 DYNA3D or NIKE3D 300 - PING
NGLBV	1	18	Number of Global State Variables
IT	1	19	State Nodal Temperatures Included? 0 - No, 1 - Yes
IU	1	20	State Nodal Coordinates Included? 0 - No, 1 - Yes
IV	1	21	State Nodal Velocities Included? 0 - No, 1 - Yes
IA	1	22	State Nodal Accelerations Included? 0 - No, 1 - Yes
*NEL*8	1	23	Number of 8-Node Hexahedral Elements
*NUMMAT*8	1	24	Number of Materials Used by Hexahedral Elements
Blank	2	25	
*NV*3*D*	1	27	Number of Variables for Hexahedral Elements
*NEL*2	1	28	Number of 2-Node Beam Elements
*NUMMAT*2	1	29	Number of Materials Used by Beam Elements
*NV*1*D*	1	30	Number of Variables for Beam Elements
*NEL*4	1	31	Number of 4-Node Shell Elements
*NUMMAT*4	1	32	Number of Materials Used by Shell Elements
*NV*2*D*	1	33	Number of Variables for Shell Elements
Blank	1	34	
ACTIV	1	35	Element Activity Data Included? 0 - No, [1000,1005] - Yes
Blank	28	36	

for all of the beam, shell and hexahedral elements. Otherwise, no element activity data is included in the plot-state section of the data base.

3.2 The Geometry and Connectivity Data Section

The geometry and connectivity data describes the initial coordinates of the nodal points, the connectivity of elements to nodal points and the material type of each element. This section occurs only once in the data base. The nodal point coordinates are floating-point numbers, while the nodal connectivities and material types are integers. This data follows the control data in the plot data base, starting at word 64. The order of this data is shown in Table 2.

3.3 The Plot-State Data Section(s)

The plot-state data section(s) contains, in general, both nodal-point and element based simulation results for *each* of the nodal points and elements which comprise the mesh. A given plot data base will usually contain several state data sections, corresponding to the various times during the simulation at which the data base was dumped. All of the entries in the state data section are floating-point numbers. The order of this data is shown in Table 3.

For data bases produced by the PING code [3] ($ICODE = 300$) the entries for the v_x, v_y and v_z velocities of the nodal points associated with fluid elements are replaced with the incident (p_{inc}), scattered (p_{scat}) and total (p_{tot}) pressures, respectively.

Element activity entries will be included in the state data section only if $1000 \leq ACTIV \leq 1005$. A single word will be output for each element, either 1.0 if the element was still active at the time that the given state was written, or 0.0 if the element had been deleted prior to the state being written.

Table 2
The Geometry and Connectivity Data Section

Data	Words	Description
Nodal Point Data	$NDIM * NUMNP$	Coordinates for each Nodal Point: (1) x Coordinate (2) y Coordinate (3) z Coordinate (Only if $NDIM = 3$)
Hex Element Data	$9 * NEL8$	Connectivity and Material for each 8-Node Hexahedral Element: (1) Nodal Point 1 (2) Nodal Point 2 (3) Nodal Point 3 (4) Nodal Point 4 (5) Nodal Point 5 (6) Nodal Point 6 (7) Nodal Point 7 (8) Nodal Point 8 (9) Material
Beam Element Data	$6 * NEL2$	Connectivity, Orientation Node, Two Blank Entries, and Material for Each 2-Node Beam Element: (1) Nodal Point 1 (2) Nodal Point 2 (3) Orientation Nodal Point (4) `Blank` (5) `Blank` (6) Material
Shell Element Data	$5 * NEL4$	Connectivity and Material for each 4-Node Shell Element: (1) Nodal Point 1 (2) Nodal Point 2 (3) Nodal Point 3 (4) Nodal Point 4 (5) Material

Table 3
The Plot-State Data Section

Data	Words	Description
Time	1	Time of the Plot State Dump.
Global Results	$NGLBV$	Integrated Global Results.
Nodal Results	NND	Results for each Nodal Point: $NND = NUMNP * (IT + NDIM * (IU + IV + IA))$ Coordinates, if $IU = 1$: (1) x Coordinate (2) y Coordinate (3) z Coordinate (Only if $NDIM = 3$) Velocities, if $IV = 1$ and $ICODE = 2$ or $ICODE = 6$: (1) x Velocity (2) y Velocity (3) z Velocity (Only if $NDIM = 3$) Pressures, if $IV = 1$ and $ICODE = 300$: (1) Incident Pressure (2) Scattered Pressure (3) Total Pressure Accelerations, if $IA = 1$: (1) x Acceleration (2) y Acceleration (3) z Acceleration (Only if $NDIM = 3$) Temperatures, if $IT = 1$: (1) Temperature

Table 3 (continued)

Data	Words	Description
Hex Results	$N8$	Results for each 8-Node Hexahedral Element: $N8 = NEL8 * NV3D$ (1) x Stress (2) y Stress (3) z Stress (4) xy Stress (5) yz Stress (6) zx Stress (7) Effective Plastic Strain
Beam Results	$N2$	Results for each 2-Node Beam Element: $N2 = NEL2 * NV1D$ (1) Axial Force (2) s Shear (3) t Shear (4) s Moment (5) t Moment (6) Torsional Moment
Shell Results	$N4$	Results for each 4-Node Shell Element: $N4 = NEL4 * NV2D$ (1) x Stress (Middle surface) (2) y Stress (Middle surface) (3) z Stress (Middle surface) (4) xy Stress (Middle surface) (5) yz Stress (Middle surface) (6) zx Stress (Middle surface) (7) Effective Plastic Strain (Middle surface) (8) x Stress (Inner surface) (9) y Stress (Inner surface) (10) z Stress (Inner surface) (11) xy Stress (Inner surface) (12) yz Stress (Inner surface) (13) zx Stress (Inner surface) (14) Effective Plastic Strain (Inner surface) (15) x Stress (Outer surface) (16) y Stress (Outer surface) (17) z Stress (Outer surface) (18) xy Stress (Outer surface) (19) yz Stress (Outer surface) (20) zx Stress (Outer surface) (21) Effective Plastic Strain (Outer surface)

Table 3 (continued)

Data	Words	Description
Shell Results	$N4$	(continued) (22) Bending Moment M_x (23) Bending Moment M_y (24) Bending Moment M_{xy} (25) Shear Resultant Q_x (26) Shear Resultant Q_y (27) Normal Moment N_x (28) Normal Moment N_y (29) Normal Moment N_{xy} (30) Thickness (31) Element Dependent Variable 1 (32) Element Dependent Variable 2 (33) Internal Energy If $NV2D = 45$: (34) x Strain (Inner surface) (35) y Strain (Inner surface) (36) z Strain (Inner surface) (37) xy Strain (Inner surface) (38) yz Strain (Inner surface) (39) zx Strain (Inner surface) (40) x Strain (Outer surface) (41) y Strain (Outer surface) (42) z Strain (Outer surface) (43) xy Strain (Outer surface) (44) yz Strain (Outer surface) (45) zx Strain (Outer surface)
Activity Data	NA	Element Activity Data, if $ACTIV = 1$. $NA = NEL8 + NEL2 + NEL4$ Hexahedral ($NEL8$) plus Beam ($NEL2$) plus Shell ($NEL4$)

4 Time-History Data Base Description

The current form of the time-history data base was originally designed to work with the TAURUS post-processor [7]. The format of this data base is also compatible for analysis with the THUG post-processor [5]. The time-history data base is similar to the plot data base described above. It is composed of at least three sections: a section of control data which provides the problem parameters, a geometry and connectivity section which defines the computational mesh, and at least one time-history state or "snapshot" of various nodal and elemental results of the simulation for a *specified subset* of the nodal points and element which form the computational mesh.

4.1 The Control Data Section

The control data section comprises *at least* the first 64 words of the time-history data base. All of the entries in the control data section are integers. This section consists of problem description parameters in the order shown in Table 4.

There are a small number of differences in the first 64 words of the control-data sections of the plot and time-history data bases. The blank entries at file addresses 25 and 26 in the plot data base are replaced with the variables $NUMDS$ and $NUMST$, which represent the number of nodal-point and element time-history blocks included in the time-history result section of the time-history data base. In addition, the blank entries at file addresses 59 to 62 in the plot data base are replaced with the variables $INEW$, $NSTH$, $NSTB$ and $NSTS$, where the first of these variables describes whether this is an old or new style data base, and the final three variables represent the number of hexahedral, beam and shell-element time-history blocks ($NUMST = NSTH + NSTB + NSTS$). Finally, the entry at file address 35 in the plot data base representing the element activity flag $ACTIV$ is replaced by a blank entry in the time-history data base.

Following the first 64 words in the data base, the nodal-point and element time-history blocks are defined. Each block is defined by the minimum and maximum nodal-point or element within the block. Therefore, starting at file address 64, there are $2 * NUMDS$ words which define the nodal point blocks, followed by $2 * NSTH$ words which define the hexahedral element blocks, then $2 * NSTB$ words that define the beam element blocks and finally $2 * NSTS$ words which define the shell element blocks.

Table 4
The Control Data Section (Time-History Data Base)

Variable	Words	Address	Description
$TITLE$	10	0	Problem Identification
Blank	5	10	
$NDIM$	1	15	Number of Dimensions: 2 - Two-Dimensional 3 - Three-Dimensional, Packed 4 - Three-Dimensional, Unpacked
$NUMNP$	1	16	Number of Nodal Points
$ICODE$	1	17	Code Descriptor: 2 - Pre-1988 DYNA3D 6 - Post-1988 DYNA3D 300 - PING
$NGLBV$	1	18	Number of Global Time-History Variables
IT	1	19	Time-History Nodal Temperatures Included? 0 - No, 1 - Yes
IU	1	20	Time-History Nodal Coordinates Included? 0 - No, 1 - Yes
IV	1	21	Time-History Nodal Velocities Included? 0 - No, 1 - Yes
IA	1	22	Time-History Nodal Accelerations Included? 0 - No, 1 - Yes
$NEL8$	1	23	Number of 8-Node Hexahedral Elements
$NUMMAT8$	1	24	Number of Materials Used by Hexahedral Elements
$NUMDS$	1	25	Number of Nodal Point Time-History Blocks
$NUMST$	1	26	Number of Element Time-History Blocks
$NV3D$	1	27	Number of Variables for Hexahedral Elements
$NEL2$	1	28	Number of 2-Node Beam Elements
$NUMMAT2$	1	29	Number of Materials Used by Beam Elements
$NV1D$	1	30	Number of Variables for Beam Elements
$NEL4$	1	31	Number of 4-Node Shell Elements
$NUMMAT4$	1	32	Number of Materials Used by Shell Elements
$NV2D$	1	33	Number of Variables for Shell Elements
Blank	25	34	
$INEW$	1	59	Time History Data Base Descriptor: $<$ 1000 - Old-Style Data Base $\geq$ 1000 - New-Style Data Base
$NSTH$	1	60	Number of Hexahedral Element Time-History Blocks
$NSTB$	1	61	Number of Beam Element Time-History Blocks
$NSTS$	1	62	Number of Shell Element Time-History Blocks
Blank	1	63	
$NDSOUT$	$2 * NUMDS$	64	Nodal Point Time-History Blocks
$NSTOUT$	$2 * NUMST$	$64 +$ $2 * NUMDS$	Element Time-History Blocks: Hexahedral ($2 * NSTH$) plus Beam ($2 * NSTB$) plus Shell ($2 * NSTS$)

4.2 The Geometry and Connectivity Data Section

The geometry and connectivity data section for the time-history data base is identical to the data that is written to the plot data base. See Table 2 for a description of the layout of the data in this section.

4.3 The Time-History Data Section(s)

The time-history data section(s) contains, in general, both nodal-point and element based simulation results for a *selected subset* the nodal points and elements which comprise the mesh. A given time-history data base will usually contain several time-history data sections, corresponding to the various times during the simulation at which the data base was dumped. All of the entries in the time-history data section are floating-point numbers. The order of this data is shown in Table 5.

There are five subsections to the time-history data section, which are organized into data associated with mesh-integrated global variables, nodal points and hexahedral, beam and shell elements. Since not all of these subsections are required in a given data base, those subsections that are included each begin with a single word which represents the simulation time at which the corresponding time-history data was dumped. Hence, it is possible to have up to five time words in a single time-history data section.

Note that the amount of nodal-point data that is written to the time-history data base assumes a three-dimensional simulation. Therefore, nine words are written per selected nodal point when IU, IV and IA are each set to 1, but only seven words are written when $IA = 0$, since a blank word is written after the v_z entry.

Also note that the hexahedral and shell element subsections each contain result data, as well as element-connected nodal point data, but that the beam element subsection contains only result data.

Table 5
The Time-History Data Section

Data	Words	Description
If $NGLBV > 0$:		
Time	1	Time of the Time-History Dump.
Global Results	$NGLBV$	Integrated Global Results.
If $NUMDS > 0$:		
Time	1	Time of the Time-History Dump.
Nodal Results	NLN	Results for $NPTH$ Time-History Nodal Points: If $IA = 0: \quad NLN = NPTH * (3 * (IU + IV) + 1)$ (1) x Displacement (2) y Displacement (3) z Displacement (4) x Velocity (5) y Velocity (6) z Velocity (7) Blank If $IA = 1: \quad NLN = NPTH * (3 * (IU + IV + IA))$ (1) x Displacement (2) y Displacement (3) z Displacement (4) x Velocity (5) y Velocity (6) z Velocity (7) x Acceleration (8) y Acceleration (9) z Acceleration
If $NSTH > 0$:		
Time	1	Time of the Time-History Dump.
Hex Nodal Point Data	$NHNPV$	Nodal Point Data for HTH Time-History Hexahedral Elements: $NHNPV = HTH * 56$ (1 - 8) Nodal Point Connectivity (9 - 32) x, y and z Coordinates of Each Connected Nodal Point (33 - 56) x, y and z Velocities of Each Connected Nodal Point

Table 5 (continued)

Data	Words	Description
Hex Element Results	NHV	Results for HTH Time-History Hexahedral Elements: $NHV = HTH * NV3D$ (1) x Stress (2) y Stress (3) z Stress (4) xy Stress (5) yz Stress (6) zx Stress (7) Effective Plastic Strain
If $NSTB > 0$:		
Time	1	Time of the Time-History Dump.
Beam Element Results	NBV	Results for BTH Time-History Beam Elements: $NBV = BTH * NV1D$ (1) Axial Force (2) s Shear (3) t Shear (4) s Moment (5) t Moment (6) Torsional Moment
If $NSTS > 0$:		
Time	1	Time of the Time-History Dump.
Shell Nodal Point Data	$NSNPV$	Nodal Point Data for STH Time-History Shell Elements: $NSNPV = HTH * 28$ (1 - 4) Nodal Point Connectivity (4 - 16) x, y and z Coordinates of Each Connected Nodal Point (17 - 28) x, y and z Velocities of Each Connected Nodal Point
Shell Element Results	NSV	Results for STH Time-History Shell Elements: $NSV = STH * NV2D$ (1) x Stress (Middle surface) (2) y Stress (Middle surface) (3) z Stress (Middle surface) (4) xy Stress (Middle surface) (5) yz Stress (Middle surface) (6) zx Stress (Middle surface) (7) Effective Plastic Strain (Middle surface) (8) x Stress (Inner surface) (9) y Stress (Inner surface) (10) z Stress (Inner surface) (11) xy Stress (Inner surface) (12) yz Stress (Inner surface) (13) zx Stress (Inner surface) (14) Effective Plastic Strain (Inner surface)

Table 5 (continued)

Data	Words	Description
Shell Element Results		(continued) (15) x Stress (Outer surface) (16) y Stress (Outer surface) (17) z Stress (Outer surface) (18) xy Stress (Outer surface) (19) yz Stress (Outer surface) (20) zx Stress (Outer surface) (21) Effective Plastic Strain (Outer surface) (22) Bending Moment M_x (23) Bending Moment M_y (24) Bending Moment M_{xy} (25) Shear Resultant Q_x (26) Shear Resultant Q_y (27) Normal Moment N_x (28) Normal Moment N_y (29) Normal Moment N_{xy} (30) Thickness (31) Element Dependent Variable 1 (32) Element Dependent Variable 2 (33) Internal Energy If $NV2D = 45$: (34) x Strain (Inner surface) (35) y Strain (Inner surface) (36) z Strain (Inner surface) (37) xy Strain (Inner surface) (38) yz Strain (Inner surface) (39) zx Strain (Inner surface) (40) x Strain (Outer surface) (41) y Strain (Outer surface) (42) z Strain (Outer surface) (43) xy Strain (Outer surface) (44) yz Strain (Outer surface) (45) zx Strain (Outer surface)

Acknowledgments

The authors wish to thank Brad Maker and Don Dovey for providing the I/O packages from TDBD (the TAURUS Data Base Decoder) and GRIZ, which served as a starting point for the develoment of the merging utility CombinePlt. We would also like to acknowledge helpful conversations with Carol Hoover and Doug Speck.

References

[1] Whirley, R.G. and Englemann, B.E., "DYNA3D: A Nonlinear, Explicit, Three-Dimensional Code for Finite Element Code for Solid and Structural Mechanics - User Manual", Report UCRL-MA-107254, Revision 1, Lawrence Livermore National Laboratory, Livermore, CA (1993).

[2] Maker, B.N., Ferencz, R.M. and Hallquist, J.O., "NIKE3D: A Nonlinear, Implicit, Three-Dimensional Code for Finite Element Code for Solid and Structural Mechanics - User Manual", Report UCRL-MA-105268, Lawrence Livermore National Laboratory, Livermore, CA (1993).

[3] Christon, M.A., "PING: An Explicit Finite Element Code for Linear Structural Acoustics - User Manual", Report UCRL-MA-114536, Lawrence Livermore National Laboratory, Livermore, CA (1993).

[4] Dovey, D. and Spelce, T.E., "GRIZ: Finite Element Analysis Results Visualization for Unstructured Grids - User Manual", Report UCRL-MA-115696, Lawrence Livermore National Laboratory, Livermore, CA (1993).

[5] Speck, D.E., "THUG: Time Histories from Unstructured Grids - User Manual", Report UCRL-MA-XXXXXX, Lawrence Livermore National Laboratory, Livermore, CA (1995).

[6] Procassini, R.J., DeGroot, A.J. and Maltby, J.D., "PartMesh User Manual - Partitioning Unstructured Finite Element Meshes for Solution on a Massively Parallel Processor", Report UCRL-MA-118774, Lawrence Livermore National Laboratory, Livermore, CA (1994).

[7] Spelce, T.E., Brown, B.E. and Hallquist, J.O., "TAURUS: An Interactive Post-Processor for the Analysis Codes NIKE3D, DYNA3D and TOPAZ3D", Report UCRL-MA-105401, Lawrence Livermore National Laboratory, Livermore, CA (1991).

Technical Information Department • Lawrence Livermore National Laboratory
University of California • Livermore, California 94550