Date June 3, 2000 Memo Number STI50:000603
Subject **ANSYS Tips & Tricks: APDL Vector functions**
Keywords APDL: Vector: *VGET

1. Introduction:

APDL (ANSYS Parametric Design Language) provides the user with flexibility and power in creating complex macros and input files.[1] Part of this includes the usage of *DO loops for repetitive tasks or to *GET information.

Vector notation/functions in ANSYS allow the user to execute some of these *DO loops in a much more efficient manner. Instead of looping through individual functions, vector functions act upon arrays, resulting in faster execution times.

This memo will provide some basic/introductory examples on the use of *VGET as well as the undocumented vector notation. Although it is assumed that the user is running 5.6, these will work in 5.5 as well.

2. Background Information on *VGET:

Users may already be familiar with the *GET command to retrieve information from the database. *GET allows the user to get input as well as output information, ranging from keypoint positions to volume of elements to nodal results.

*VGET has similar functionality, but it acts upon an arrays rather than a single parameter. *VGET does not support as many options as *GET since it only is used for vector quantities, such as DOF results or element attributes of volumes.

Because *VGET acts upon a vector, *VMASK usually is used in conjunction with *VGET. *VMASK is a masking array to tell ANSYS to perform vector operations only on certain items in the array. For example, a user may have 200 nodes, but only 30 nodes selected. *VMASK will tell ANSYS to perform the operation only on 30 nodes (array entries) which are selected.

To use *VGET, one needs to perform the following steps:

1) Define a masking array with *DIM
2) Define a regular array with *DIM to hold data of interest
3) Fill masking array with selected nodes, using *VGET,parm,,node,1,nsel[2]
4) Activate masking array with *VMASK
5) Fill regular array with quantity of interest using *VGET

A simple example is shown below:

```
<select nodes of interest after solving a problem>
*get,NNUMMAX,node,,num,max              ! Get max node number
*del,NMASK                              ! Delete NMASK array, if it exists
*del,NARRAY                             ! Delete NARRAY array, if it exists
*dim,NMASK,array,NNUMMAX                ! Define NMASK array
*dim,NARRAY,array,NNUMMAX,4                    ! Define NARRAY array to hold results
*vget,NMASK(1),node,1,nsel              ! Get status of selected nodes
                                        ! 1 = selected
                                        ! 0 = undefined, -1 = unselected
*vmask,NMASK(1)                         ! Use NMASK as masking array
                                        ! if NMASK(i) > 0.0, perform operation "i"
                                        ! if NMASK(i) < 0.0, do not perform on "i"
*vget,NARRAY(1,1),node,1,u,x            ! Get UX for nodes only if selected
*vmask,NMASK(1)                         ! Reactivate masking for next operation
                                        ! *VMASK only works per command
*vget,NARRAY(1,2),node,2,u,y            ! GET UY for nodes only if selected
*vmask,NMASK(1)
*vget,NARRAY(1,3),node,3,u,z            ! GET UZ for nodes only if selected
*vfil,NARRAY(1,4),ramp,1,1              ! Fill vector from 1 to NNUMMAX (node no.)
```

[1] Please refer to the ANSYS APDL Programmer's Guide in the online help or to CSI's Tip of the Week on APDL Macros by M. Rife for additional information on APDL.
[2] There exist analogous *VGET commands for elements, keypoints, lines, areas, and volumes.

```
*cfopen,vmask,txt                          ! Open file called "vmask.txt"
*vwrite                                    ! Write header information
('NODE',10x,'UX',10x,'UY',10x,'UZ')
*vmask,NMASK(1)                            ! Use mask, then write only selected nodes
*vwrite,NARRAY(1,4),NARRAY(1,1),NARRAY(1,2),NARRAY(1,3)
(F10.0,tl1,' ',F10.6,' ',F10.6,' ',F10.6)
*cfclos                                    ! Close "vmask.txt" file
*uilist,vmask.txt                          ! Pop up vmask.txt file to user
```

The above serves as an elementary use of *VGET and *VMASK to write specific data to a text file – this could be done, for example, to write nodal information to a secondary post-processing program such as spreadsheet software like Excel or Tecplot.

Please note that *VMASK only works on the next command, so it needs to be "reactivated" for multiple commands.

Also, it is instructive to remember that while DOF solution is stored at all of the nodes of an element, element solution items such as stresses are not stored at the midside nodes of higher-order elements. Be sure to use NSLE,S,CORNER to select only corner nodes prior to using *VGET & *VMASK. Solution printout (PRxxxx family of commands) with /GRAPH,POWER and /EFACET,2 *will* show midside node stresses.

3. Discussion of Vector Commands:

Vector notation/commands allow a user to execute certain commands for an array, rather than using *DO loops or the *REPEAT command. Vector notation involves using a regular ANSYS command but replacing certain fields with an array. The vector notation is (ISTART:IFINISH:IINCR) where ISTART is the starting number, IFINISH is the ending number, and IINCR is the increment (default is increment of 1).

For example, assuming that we want to define element type 1 through 6 to be PLANE42, we could use:

```
ET,(1:6),42
```

instead of a *DO loop or repeated commands. Similarly, if a user wants to define multiple contact pairs with different real constants, he/she could use the following:

```
ET,(8:14:2),170
ET,(9:15:2),173
R,(9:15:2),
```

which creates element types 8, 10, 12, and 14 for the target surfaces, element types 9, 11, 13, and 15 for the contact surfaces, and real constants 9, 11, 13, 15 for the four contact pairs. These illustrate some very simple uses of vector notation.

An argument to a command could also be a previously defined array. Consider the case where an array of X, Y, and Z locations of 400 keypoints have been defined in arrays X(1), Y(1), and Z(1), respectively. To generate the keypoints, one could use:

```
K,(1:400),X(1:400),Y(1:400),Z(1:400)
```

Commands such as node creation (N) and material property definition (MP) also support vector notation.

Vector notation is an undocumented feature of ANSYS, so please use this at your own discretion. While it is not officially supported by ANSYS, Inc., CSI will provide as much help/guidance to its customers regarding this functionality. The author uses vector notation often without any problems.

4. Benefits of Using *VGET and Vector notation:

One can see that *VGET and vector notation offer the user much more flexibility than the *REPEAT command. Also, use of vector notation can significantly reduce the size of a macro or input file, resulting in easier management and modification of macro files.

Besides easier programming, leading to "leaner" input files, execution of vector notation is much faster than using *DO loops. This is because the command acts upon a vector rather than individual items. To demonstrate this, three files have been provided:

1) buildn.inp: Input file for demonstration
2) buildn1.mac: Macro to scale nodes using *DO loops
3) buildn2.mac: Macro to scale nodes using *VGET and vector notation

The above "buildn.inp" input file creates a simple block, meshes it, detaches the mesh from the volume, then scales the nodes and elements by a factor of 2. [*Please note that, generally speaking, the author does not recommend detaching the finite element mesh from the solid model! This was done in this macro for illustrative purposes only!*]

On a Dell Inspiron 7500 running a 500 MHz Intel processor, the author got the following results:

```
Using DO Loops
cpu      83.439977 seconds   wall     84.000000 seconds

Using Vector functions
cpu      17.855667 seconds   wall     18.000000 seconds
```

One can clearly see from the results that use of vector notation/functions result in significant time savings, in this case, over 4.5 times faster.

5. Conclusion:

Although this memo is meant to introduce *VGET and vector notation to the user, there are many more capabilities not covered presently. The user is instructed to refer to the APDL Programmer's Guide for more details on *VOPER, *MOPER, and other array-manipulation commands.

While *VGET and vector notation cannot be used for every situation, the author hopes that the use of these commands, when applicable, may help the user create more efficient, faster macros to help them become more productive using ANSYS.

6. References:

Swanson, John, "Programming in ANSYS" Course notes, 8/16/98

Sheldon Imaoka
Collaborative Solutions, Inc. (LA Office)
Engineering Consultant

ANSYS Tip of the Week

"ANSYS Tip of the Week" (TOTW) is provided for customers of Collaborative Solutions, Inc. (CSI) with active TECS agreements, distributed weekly in Adobe Acrobat PDF format via email. Unless otherwise stated, information contained herein should be applicable to ANSYS 5.4 and above, although usage of the latest version (5.6 as of this writing) is assumed. Users who wish to subscribe/unsubscribe or to view older TOTW archives can visit

http://www.csi-ansys.com/tip_of_the_week.htm

Corrections, comments, and suggestions are welcome and can be sent to operator@csi-ansys.com [they will be distributed to the appropriate person(s)]. While CSI engineers base their TOTW on technical support calls and user questions, ideas on future topics are appreciated. Users who wish to submit their own TOTW are encouraged to do so by emailing the above address for more information.

XANSYS Mailing List

The XANSYS mailing list is a forum for questions and discussions of the use of ANSYS. As of 04/00, there are more than 1000 subscribers with topics ranging from Structural, Thermal, Flotran, to Emag analyses, to name a few. Users are encouraged to subscribe to evaluate the usefulness of the mailing list for themselves. Also, either (a) using the mail program to filter [xansys] messages or (b) using the "digest" option to receive one combined email a day is strongly recommended to minimize sorting through the volume of postings.

```
This list is for *ALL* users of the ANSYS finite element analysis program from around the
world. The list allows rapid communication among users concerning program
bugs/ideas/modeling techniques. This list is NOT affiliated with ANSYS, Inc. even though
several members of the ANSYS, Inc. staff are subscribers and regular contributors.

To SUBSCRIBE: send blank email to xansys-subscribe@onelist.com
To unsubscribe send blank email to xansys-unsubscribe@onelist.com
Archived on http://www.infotech.tu-chemnitz.de/~messtech/ansys/ansys.html
ANOTHER archive on http://www.eScribe.com/software/xansys/
(A poor archive is also at http://www.onelist.com/archives.cgi/xansys)
```

CSI ANSYS Technical Support, Training, & Mentoring

Collaborative Solutions, Inc. is committed to providing the best customer support in our industry. Three people will be devoted to technical support from 8:00 a.m. to 5:00 p.m. PST every working day. CSI customers with active TECS (maintenance) agreements may contact CSI by any of the following ways:

Phone: 760-431-4815
Fax: 760-431-4824
E-mail: firstname.lastname@csi-ansys.com

WWW: http://www.csi-ansys.com
FTP: ftp://ftp.csi-ansys.com

CSI Engineers:

Karen Dhuyvetter	Greg Miller	Sean Harvey	Alfred Saad
Bill Bulat	Sheldon Imaoka	David Haberman	Mike Rife

CSI believes strongly in the value of training and mentoring to help make customers successful using ANSYS. Training classes are usually 2-3 days in duration and provide instruction on various topics, including structural nonlinearities, heat transfer, and dynamics. Mentoring sessions involve working with a CSI engineer one-on-one on specific projects. These sessions help reinforce applicable subject matter covered in training classes or help ensure that the customer is using ANSYS most efficiently and effectively.

Training class schedules are posted at: http://www.csi-ansys.com/training.htm

Please contact your account manager for more details on training, mentoring, consulting, and other CSI services.