AFNI program: ConvertDset

Output of `-help`

```
Usage:
  ConvertDset -o_TYPE -input DSET [-i_TYPE] [-prefix OUT_PREF]
  Converts a surface dataset from one format to another.

  Mandatory parameters:
     -o_TYPE: TYPE of output datasets
              where TYPE is one of:
           niml_asc (or niml): for ASCII niml format.
           niml_bi:            for BINARY niml format.
           1D:                 for AFNI's 1D ascii format.
           1Dp:                like 1D but with no comments
                               or other 1D formatting gimmicks.
           1Dpt:               like 1Dp but transpose the output.
           gii:                GIFTI format default .
           gii_asc:            GIFTI format with ascii DataArrays.
           gii_b64:            GIFTI format with Base 64 encoded DataArrays.
           gii_b64gz:          GIFTI format with B64 encoding and gzipping.
         For stderr and stdout output use one of:
           1D_stderr, 1D_stdout, niml_stderr, or niml_stdout,
           1Dp_stdout, 1Dp_stderr, 1Dpt_stdout, 1Dpt_stderr
         Actually, this parameter is not that mandatory, the program
         can look at extensions on the prefix to guess the output
         format. If the prefix has no extension and o_TYPE is not
         specified, then the output format is the same as that of the
         input.
     -input DSET: Input dataset to be converted.
                  See more on input datasets below.
     -dset_labels 'SB_LABEL_0 SB_LABEL_1 ...'
                  Label the columns (sub-bricks) of the output dataset
                  You must have as many labels as you have sub-bricks in
                  the output dataset.  Optional parameters:
     -add_node_index: Add a node index element if one does not exist
                      in the input dset. With this option, the indexing
                      is assumed to be implicit (0,1,2,3.... for rows 0,1
                      2,3,...). If that is not the case, use -node_index_1D
                      option below.
              ** NOTE: It is highly recommended you use one of -node_index_1D
                       or -add_node_index when going from 1D format to NIML
                       GIFTI formats.
     -node_index_1D INDEX.1D: Specify file containing node indices
                              Use this to provide node indices with
                              a .1D dset. In many cases for .1D data
                              this option is DSET.1D'[0]'
     -node_select_1D MASK.1D: Specify the nodes you want to keep in the
                              output.
                              The order of the rows in the output dataset
                              reflects the order of the nodes in MASK.1D.
                              Note that the presence of duplicate nodes in
                              MASK.1D is not allowed, so if MASK.1D came
                              from ROI2dataset's -nodelist, recreate it with
                              option -nodelist.nodups instead.
                              Also, node indices that do not have data in the
                              input dataset will be ignored.
```

```
                        When in doubt, use the 1D output format along
                        with -prepend_node_index_1D and spot check your
                        results.
-prepend_node_index_1D: Add a node index column to the data, rather
                        than keep it as part of the metadata.
-pad_to_node MAX_INDEX: Output a full dset from node 0
                       to node MAX_INDEX (a total of
                       MAX_INDEX + 1 nodes). Nodes that
                       get no value from input DSET are
                       assigned a value of 0
                       If MAX_INDEX is set to 0 it means you want
                       to pad the maximum node in the input dataset.
       ** Notice that padding gets done at the very end.
       ** Instead of directly setting MAX_INDEX to an integer you
          can set MAX_INDEX to something like:
       ld120 (or rd17) which sets MAX_INDEX to be the maximum
          node index on an Icosahedron with -ld 120. See
          CreateIcosahedron for details.
       d:DSET.niml.dset which sets MAX_INDEX to the maximum node found
                in dataset DSET.niml.dset.

-labelize CMAP: Turn the dataset into a labeled set per the colormap in
                CMAP. A CMAP can easily be generated with MakeColorMap's
                options -usercolorlutfile and -suma_cmap.

-graphize: Turn the dataset into a SUMA graph dataset.
           See input format constraints under -onegraph and -multigraph
-graph_nodelist_1D NODEINDLIST.1D NODELIST.1D: Two files specifying the
                            indices and the coordinates of the graph's
                            nodes. In sum you need I X Y Z (RAI mm).
                            but the I comes from NODEINDLIST.1D and the
                            X Y Z coordinates from NODELIST.1D
                            If you have everything in one file, use
                            the same filename twice with proper column
                            selectors.
-graph_full_nodelist_1D NODELIST.1D: Same as above, but without the need
                            for NODEINDLIST.1D. In that case, indices
                            will implicitly go from 0 to N-1, with N
                            being the number of nodes.
-graph_named_nodelist_txt NODENAMES.txt NODELIST.1D: Two files used to
                            specify graph node indices, string labels,
                            and their coordinates.
                            In sum you need I LABEL X Y Z (RAI mm).
                            The I and LABEL come from NODENAMES.txt and
                            the X Y Z coordinates from NODELIST.1D
                      Also, you can assign to each graph node a group ID
                            and nodes with the same group ID can be
                            displayed with the same color in SUMA.
                            To do so, add a third column to
                            NODENAMES.txt so that you have: I LABEL GID
                            with GID being the integer group ID.
                            Color selection for the different group IDs
                            is done automatically with ConvertDset, but
                            you can set your own by appending three
                            more columns to NODENAMES.txt to have:
                               I LABEL GID R G B
                            with R, G, and B values between 0 and 1.0
-graph_XYZ_LPI: Coords in NodeList.1D are in LPI instead of RAI
-graph_edgelist_1D EDGELIST.1D: i j indices of graph nodes defining edge
                             with each row matching the input dset row.
                             This option only works with -multigraph
                             This option also marks the graph as being
```

```
                                a sparse matrix, even if a square matrix
                                is provided.
    -onegraph: Expect input dataset to be one square matrix defining the
               graph (default).
    -multigraph: Expect each column in input dataset to define an entire
                 graph. Each column in this case should be a column-stacked
                 square matrix.

    -i_TYPE: TYPE of input datasets
             where TYPE is one of:
          niml: for niml data sets.
          1D:   for AFNI's 1D ascii format.
          dx: OpenDX format, expects to work on 1st
              object only.
          If no format is specified, the program will
          guess using the extension first and the file
          content next. However the latter operation might
          slow operations down considerably.
    -prefix OUT_PREF: Output prefix for data set.
                      Default is something based
                      on the input prefix.
    -split N: Split a multi-column dataset into about N output datasets
              with all having the same number of columns, except perhaps
              for the last one. Confused? try:
              ConvertDset -i v2s.lh.TS.niml.dset -split 3 \
                          -prefix Split3
              3dinfo -n4 -label Split3.000* v2s.lh.TS.niml.dset\
    -no_history: Do not include a history element in the output
 Notes:
    -This program will not overwrite pre-existing files.
    -The new data set is given a new idcode.

 SUMA dataset input options:
     -input DSET: Read DSET1 as input.
                  In programs accepting multiple input datasets
                  you can use -input DSET1 -input DSET2 or
                  input DSET1 DSET2 ...
      NOTE: Selecting subsets of a dataset:
            Much like in AFNI, you can select subsets of a dataset
            by adding qualifiers to DSET.
          Append #SEL# to select certain nodes.
          Append [SEL] to select certain columns.
          Append {SEL} to select certain rows.
          The format of SEL is the same as in AFNI, see section:
          'INPUT DATASET NAMES' in 3dcalc -help for details.
          Append [i] to get the node index column from
                     a niml formatted dataset.
          *  SUMA does not preserve the selection order
             for any of the selectors.
             For example:
             dset[44,10..20] is the same as dset[10..20,44]
             Also, duplicate values are not supported.
             so dset[13, 13] is the same as dset[13].
             I am not proud of these limitations, someday I'll get
             around to fixing them.


SUMA mask options:
     -n_mask INDEXMASK: Apply operations to nodes listed in
                          INDEXMASK  only. INDEXMASK is a 1D file.
     -b_mask BINARYMASK: Similar to -n_mask, except that the BINARYMASK
                         1D file contains 1 for nodes to filter and
```

```
                        0 for nodes to be ignored.
                        The number of rows in filter_binary_mask must be
                        equal to the number of nodes forming the
                        surface.
      -c_mask EXPR: Masking based on the result of EXPR.
                    Use like afni's -cmask options.
                    See explanation in 3dmaskdump -help
                    and examples in output of 3dVol2Surf -help
      NOTE: Unless stated otherwise, if n_mask, b_mask and c_mask
            are used simultaneously, the resultant mask is the intersection
            (AND operation) of all masks.


   [-novolreg]: Ignore any Rotate, Volreg, Tagalign,
                or WarpDrive transformations present in
                the Surface Volume.
   [-noxform]: Same as -novolreg
   [-setenv "'ENVname=ENVvalue'"]: Set environment variable ENVname
               to be ENVvalue. Quotes are necessary.
             Example: suma -setenv "'SUMA_BackgroundColor = 1 0 1'"
               See also options -update_env, -environment, etc
               in the output of 'suma -help'
  Common Debugging Options:
   [-trace]: Turns on In/Out debug and Memory tracing.
             For speeding up the tracing log, I recommend
             you redirect stdout to a file when using this option.
             For example, if you were running suma you would use:
             suma -spec lh.spec -sv ... > TraceFile
             This option replaces the old -iodbg and -memdbg.
   [-TRACE]: Turns on extreme tracing.
   [-nomall]: Turn off memory tracing.
   [-yesmall]: Turn on memory tracing (default).
  NOTE: For programs that output results to stdout
    (that is to your shell/screen), the debugging info
    might get mixed up with your results.


Global Options (available to all AFNI/SUMA programs)
  -h: Mini help, at time, same as -help in many cases.
  -help: The entire help output
  -HELP: Extreme help, same as -help in majority of cases.
  -h_view: Open help in text editor. AFNI will try to find a GUI editor
  -hview : on your machine. You can control which it should use by
           setting environment variable AFNI_GUI_EDITOR.
  -h_web: Open help in web browser. AFNI will try to find a browser.
  -hweb : on your machine. You can control which it should use by
          setting environment variable AFNI_GUI_EDITOR.
  -h_find WORD: Look for lines in this programs's -help output that match
                (approximately) WORD.
  -h_raw: Help string unedited
  -h_spx: Help string in sphinx loveliness, but do not try to autoformat
  -h_aspx: Help string in sphinx with autoformatting of options, etc.
  -all_opts: Try to identify all options for the program from the
             output of its -help option. Some options might be missed
             and others misidentified. Use this output for hints only.


Examples:
1-   Plot a node's time series from a niml dataset:
     ConvertDset -input DemoSubj_EccCntavir.niml.dset'#5779#' \
                 -o_1D_stdout | 1dplot -nopush -stdin
```

```
2-   Change a dataset to a labeled dataset using the colormap generated
     in Example 5 of MakeColorMap's help
        ConvertDset -i you_look_marvellous.niml.dset \
                    -o you_look_labeled.niml.dset -labelize toylut.niml.cmap
     The advantage of having a labeled dataset is that you can see the label
     of a node when you click on it in SUMA, and you can extract
     regions based on their labels. For example, with the dataset created
     above you can run the following command to extract a mask of the
     nodes labeled 'Small_Face' with something like:
        3dcalc -a you_look_labeled.niml.dset'<Small_Face>' \
               -expr 'a' -prefix Small_Face.only
     This assumes of course that your colormap toylut.niml.cmap does have
     an entry labeled 'Small_Face'



Compile Date:
   Sep 29 2023

    Ziad S. Saad SSCC/NIMH/NIH saadz@mail.nih.gov
```

This page auto-generated on Fri Sep 29 12:59:19 EDT 2023