GNU Datamash

for version 1.8, 10 July 2022

GNU Datamash Developers (assafgordon@gmail.com)

This manual is for GNU Datamash (version 1.8, 10 July 2022), which provides command-line computations on input files.

Table of Contents

1 Overview

The `datamash` program (`https://www.gnu.org/software/datamash`) performs calculation (e.g. *sum,*, *count*, *min*, *max*, *skewness*, *standard deviation*) on input files.

Example: sum up the values in the first column of the input:

```
$ seq 10 | datamash sum 1
55
```

`datamash` can group input data and perform operations on each group. It can sort the file, and read header lines.

Example: Given a file with three fields (name, subject, score), find the average score in each subject:

```
$ cat scores.txt
Name        Subject          Score
Bryan       Arts             68
Isaiah      Arts             80
Gabriel     Health-Medicine  100
Tysza       Business         92
Zackery     Engineering      54
...

$ datamash --sort --headers --group 2 mean 3 sstdev 3 < scores.txt
GroupBy(Subject)   mean(Score)   sstdev(Score)
Arts               68.9474       10.4215
Business           87.3636       5.18214
Engineering        66.5385       19.8814
Health-Medicine    90.6154       9.22441
Life-Sciences      55.3333       20.606
Social-Sciences    60.2667       17.2273
```

`datamash` is designed for interactive exploration of textual data and for automating tasks in shell scripts.

`datamash` has a rich set of statistical functions to quickly assess information in textual input files. An example of calculating basic statistic (mean, 1st quartile, median, 3rd quartile, IQR, sample-standard-deviation, and p-value of Jarque-Bera test for normal distribution:

```
$ datamash -H mean 1 q1 1 median 1 q3 1 iqr 1 sstdev 1 jarque 1 < FILE
mean(x)   q1(x)  median(x)  q3(x)   iqr(x)  sstdev(x)  jarque(x)
45.32     23     37         61.5    38.5    30.4487    8.0113-09
```

2 Invoking datamash

The format for running the `datamash` program is:

```
datamash [option]... op1 column1  [op2 column2 ...]
```

Where *op1* is the operation to perform on the values in *column1*. `datamash` reads input from stdin and performs one or more operations on the input data. If `--group` is used, each operation is performed on every group. If `--group` is not used, each operation is performed on all the values in the input file.

The `LC_NUMERIC` locale specifies the decimal-point character and the thousands separator.

`datamash` supports the following operations:

Primary operations:
: `groupby`, `crosstab`, `transpose`, `reverse`, `check`

Line-Filtering operations:
: `rmdup`

Per-Line operations:
: `base64`, `debase64`, `md5`, `sha1`, `sha224`, `sha256`, `sha384`, `sha512`, `bin`, `strbin`, `round`, `floor`, `ceil`, `trunc`, `frac`, `dirname`, `basename`, `extname`, `barename`, `getnum`, `cut`, `echo`

Group-by Numeric operations:
: `sum`, `min`, `max`, `absmin`, `absmax`, `range`

Group-by Textual/Numeric operations:
: `count`, `first`, `last`, `rand`, `unique`, `uniq`, `collapse`, `countunique`

Group-by Statistical operations:
: `mean`, `geomean`, `harmmean`, `mode`, `median`, `q1`, `q3`, `iqr`, `perc`, `antimode`, `pstdev`, `sstdev`, `pvar`, `svar`, `ms`, `rms`, `mad`, `madraw`, `sskew`, `pskew`, `skurt`, `pkurt`, `jarque`, `dpo`, `scov`, `pcov`, `spearson`, `ppearson`

Grouping options:

`--skip-comments`
`-C`
: Skip comment lines (starting with '#' or ';' and optional whitespace).

`--full`
`-f`
: Print entire input line before op results (default: print only the grouped keys). While using this option with non-linewise operations was historically permitted, it never produced very sensible output. Such usage has been deprecated, and in a future release it will result in an error.

`--group=`*`X[,Y,X]`*
`-g` *`X[,Y,X]`*
: Group input via fields *X[,Y,Z]*. By default, fields are separated by TABs. Use `--field-separator` to change the delimiter character. Input file must

be sorted by the same fields *X[,Y,Z]*. Use `--sort` to automatically sort the input. If `--group` is not specified, each operation is performed in the entire input file.

`--header-in`
: Indicates the first input line is column headers, and should not be used for any calculations.

`--header-out`
: Print column headers as first line. If the column header names are known (i.e. the input file had a header line, and the `command` was invoked with `--header-in`, `-H` or `--headers`), prints the operation and the name of the field (e.g. '`mean(X)`'). Otherwise, prints the number operation and the field number (e.g. '`mean(field-3)`').

`--headers`
`-H`
: Same as '`--header-in --header-out`'. A short option indicating the input file has a header line, and the output should contain a header line as well.

`--ignore-case`
`-i`
: Ignore upper/lower case when comparing text for grouping, sorting, and comparing unique values in the '`countunique`' and '`unique`' (or '`uniq`') operations.

`--sort`
`-s`
: Sort the input before grouping. `datamash` requires sorted input. If the input is not sorted, using `--sort` will automatically sort the input before processing it further. Sorting will be performed based on the specified `--group` parameter, and respecting case `--ignore-case` option (if used). The following commands are equivalent:

```
$ cat FILE | sort -k1,1 | datamash --group 1 sum 1
$ cat FILE | datamash --sort --group 1 sum 1
```

`--sort-cmd=`*`PATH`*
: Use the given program to sort instead of the system `sort`

File Operation options:

`--no-strict`
: Allow lines with varying number of fields. By default, `transpose` and `reverse` will fail with an error message unless all input lines have the same number of fields.

`--filler=x`
: When use `--no-strict` option, missing fields will be filled with this value.

General options:

`--format=`*`FORMAT`*
: print numeric values with printf style floating-point *FORMAT*.

`--field-separator=`*x*
`-t` *x*
: Use character *X* instead of TAB as input and output field delimiter. If `--output-delimiter` is also used, it will override the output field delimiter.

`--narm`
: Skip *NA* or *NaN* values.

`--output-delimiter=`*x*
: Use character *X* instead as output field delimiter. This option overrides `--field-separator`/`-t`/ `--whitespace`/`-W`.

`--collapse-delimiter=`*x*
`-c` *x*
: Use character *X* instead of comma to delimit items in a '`collapse`' or '`unique`' (aka '`uniq`') list.

`--round=`*N*
`-R` *N*
: Round numeric output to *N* decimal places.

`--whitespace`
`-W`
: Use whitespace (one or more spaces and/or tabs) for field delimiters. Leading whitespace is ignored, trailing whitespace results in an empty field. TAB character will be used as output field separator. If `--output-delimiter` is also used, it will override the output field delimiter.

`--zero-terminated`
`-z`
: End lines with a 0 byte, not newline.

`--help`
: Print an informative help message on standard output and exit successfully.

`--version`
: Print the version number and licensing information of Datamash on standard output and then exit successfully.

3 Available operations in `datamash`

Primary operations:

`groupby` alternative syntax for `--group`

`crosstab` cross-tabulate two fields (also known as 'pivot-tables')

`transpose`
transpose rows, columns of a text file

`reverse` reverse fields in each line of a text file

`check` verify tabular structure of input (ensure same number of fields in all lines)

Line-Filtering operation:

`rmdup` remove lines with duplicated key value

Per-Line operations:

`base64` encode the field as base64

`debase64` decode the field as base64. Exit with an error if the field is invalid base64 value which cannot be decoded.

`md5` calculates md5 hash of the field

`sha1` calculates sha1 hash of the field

`sha224` calculates sha224 hash of the field

`sha256` calculates sha256 hash of the field

`sha384` calculates sha384 hash of the field

`sha512` calculates sha512 hash of the field

`dirname` extracts the directory name of the field (assuming the field is a file name). Similar to `dirname(1)`.

`basename` extracts the base file name of the field (assuming the field is a file name). Similar to `basename(1)`.

`extname` extracts the extension of the file name of the field (assuming the field is a file name).

`extname` extracts the base file name of the field without the extension (assuming the field is a file name).

`getnum` extract a number from the field. `getnum` accepts an optional single letter option '`n/i/d/p/h/o`' affecting the detected value.

`cut` copy input field to output field (similar to `cut(1)`). When the `cut` operation is given a list of fields, the fields are copied in the given order (in contrast to `cut(1)`).

`echo` an alias for `cut`.

Group-by Numeric operations:

`sum` sum the of values

`min` minimum value

`max` maximum value

`absmin` minimum of the absolute values

`absmax` maximum of the absolute values

`range` range of values (maximum - minimum)

Group-By Textual/Numeric operations:

`count` count number of elements in the group

`first` the first value of the group

`last` the last value of the group

`rand` one random value from the group

`unique` comma-separated sorted list of unique values

`uniq` an alias for `unique`.

`--collapse-delimiter` can be used to use a different character than comma.

`collapse` comma-separated list of all input values

`--collapse-delimiter` can be used to use a different character than comma.

`countunique`
number of unique/distinct values

Group-By Statistical operations:

`mean` mean of the values

`geomean` geometric mean of the values

`harmmean` harmonic mean of the values

`trimmean` trimmed mean of the values

`ms` mean square of the values

`rms` root mean square of the values

`median` median value

`q1` 1st quartile value

`q3` 3rd quartile value

`iqr` inter-quartile range

`perc` percentile value

`mode` mode value (most common value)

`antimode`	anti-mode value (least common value)
`pstdev`	population standard deviation
`sstdev`	sample standard deviation
`pvar`	population variance
`svar`	sample variance
`mad`	Median Absolute Deviation, scaled by a constant 1.4826 for normal distributions
`madraw`	Median Absolute Deviation, unscaled
`sskew`	skewness of the (sample) group
`pskew`	skewness of the (population) group
`skurt`	Excess Kurtosis of the (sample) group
`pkurt`	Excess Kurtosis of the (population) group
`jarque`	p-value of the Jarque-Beta test for normality
`dpo`	p-value of the D'Agostino-Pearson Omnibus test for normality.

4 Statistical Operations

Equivalent R functions

GNU Datamash is designed to closely follow R project's (`https://www.r-project.org/`) statistical functions. See the `files/operators.R` file for the R equivalent code for each of datamash's operators. When building `datamash` from source code on your local computer, operators are compared to known results of the equivalent R functions.

5 Usage Examples

5.1 Summary Statistics

The following are examples of using `datamash` to quickly calculate summary statistics. The examples will use a file with three fields (name, subject, score) representing grades of students:

```
$ cat scores.txt
Shawn     Arts  65
Marques   Arts  58
Fernando  Arts  78
Paul      Arts  63
Walter    Arts  75
...
```

Counting how many students study each subject (*subject* is the second field in the input file, thus `groupby 2`):

```
$ datamash --sort groupby 2 count 2 < scores.txt
Arts            19
Business        11
Engineering     13
Health-Medicine 13
Life-Sciences   12
Social-Sciences 15
```

Similarly, find the minimum and maximum score in each subject:

```
$ datamash --sort groupby 2 min 3 max 3 < scores.txt
Arts             46      88
Business         79      94
Engineering      39      99
Health-Medicine  72     100
Life-Sciences    14      91
Social-Sciences  27      90
```

find the mean and (population) standard deviation in each subject:

```
$ datamash --sort groupby 2 mean 3 pstdev 3 < scores.txt
Arts              68.947  10.143
Business          87.363   4.940
Engineering       66.538  19.101
Health-Medicine   90.615   8.862
Life-Sciences     55.333  19.728
Social-Sciences   60.266  16.643
```

Find the median, first, third quartiles and the inter-quartile range in each subject:

```
$ datamash --sort groupby 2 median 3 q1 3 q3  3 iqr 3  < scores.txt
Arts              71      61.5      75.5     14
Business          87      83        92        9
Engineering       56      51        83       32
```

```
Health-Medicine   91      84       100       16
Life-Sciences     58.5    44.25     67.75    23.5
Social-Sciences   62      55        70.5     15.5
```

See Section 5.2 [Header Lines and Column Names], page 10, for examples of dealing with header lines.

5.2 Header Lines and Column Names

Output Header Lines

If the input does *not* have a header line, use `--header-out` to add a header in the first line of the output, indicating which operation was performed:

```
$ datamash --sort --header-out groupby 2 min  3 max 3 < scores.txt
GroupBy(field-2)  min(field-3)  max(field-3)
Arts              46            88
Business          79            94
Engineering       39            99
Health-Medicine   72           100
Life-Sciences     14            91
Social-Sciences   27            90
```

Skipping Input Header Lines

If the input has a header line (first line containing column names), use `--header-in` to skip the line:

```
$ cat scores_h.txt
Name      Major   Score
Shawn     Arts    65
Marques   Arts    58
Fernando  Arts    78
Paul      Arts    63
...


$ datamash --sort --header-in groupby 2 mean 3 < scores_h.txt
Arts             68.947
Business         87.363
Engineering      66.538
Health-Medicine  90.615
Life-Sciences    55.333
Social-Sciences  60.266
```

If the header line is not skipped, `datamash` will show an error (due to strict input validation):

```
$ datamash groupby 2 mean 3 < scores_h.txt
datamash: invalid numeric value in line 1 field 3: 'Score'
```

Using Header Lines

Column names in the input header lines can be printed in the output header lines by using `--headers` (or `-H`, both are equivalent to `--header-in --header-out`):

```
$ datamash --sort --headers groupby 2 mean 3 < scores_h.txt
GroupBy(Major)    mean(Score)
Arts              68.947
Business          87.363
Engineering       66.538
Health-Medicine   90.615
Life-Sciences     55.333
Social-Sciences   60.266
```

Or in short form (`-sH` instead of `--sort --headers`), equivalent to the above command:

```
$ datamash -sH groupby 2 mean 3
```

Column Names

When the input file has a header line, column names can be used instead of column numbers. In the example below, *Major* is used instead of the value 2, and *Score* is used instead of the value 3:

```
$ datamash --sort --headers groupby Major mean Score < scores_h.txt
GroupBy(Major)    mean(Score)
Arts              68.947
Business          87.363
Engineering       66.538
Health-Medicine   90.615
Life-Sciences     55.333
Social-Sciences   60.266
```

`datamash` will read the first line of the input, and deduce the correct column number based on the given name. If the column name is not found, an error will be printed:

```
$ datamash --sort --headers groupby 2 mean Foo  < scores_h.txt
datamash: column name 'Foo' not found in input file
```

Field names must be escaped with a backslash if they start with a digit or contain special characters (dash/minus, colons, commas). Note the interplay between escaping with backslash and shell quoting. The following equivalent command sum the values of a field named '`FOO-BAR`':

```
$ datamash -H sum FOO\\-BAR < input.txt
$ datamash -H sum 'FOO\-BAR' < input.txt
$ datamash -H sum "FOO\\-BAR" < input.txt
```

5.3 Field Delimiters

`datamash` uses tabs (ASCII character 0x09) as default field delimiters. Use `-W` to treat one or more consecutive whitespace characters as field delimiters. Use `-t`, `--field-separator` to set a custom field delimiter.

The following examples illustrate the various options.

By default, fields are separated by a single tab. Multiple tabs denotes multiple fields (this is consistent with GNU coreutils' `cut`):

```
$ printf '1\t\t2\n' | datamash sum 3
2
$ printf '1\t\t2\n' | cut -f3
2
```

Every tab separates two fields. A line starting with a tab thus starts with an empty field, and a line ending with a tab ends with an empty field.

Using `-W`, one or more consecutive whitespace characters are treated as a single field delimiter:

```
$ printf '1  \t  2\n' | datamash -W sum 2
2
$ printf '1  \t  2\n' | datamash -W sum 3
datamash: invalid input: field 3 requested, line 1 has only 2 fields
```

With `-W`, leading whitespace is ignored, but trailing whitespace is significant. A line starting with one or more consecutive whitespace characters followed by a non-whitespace character starts with a non-empty field. A line ending with one or more consecutive whitespace characters ends with an empty field.

Using `-t`, a custom field delimiter character can be specified. Multiple consecutive delimiters are treated as multiple fields:

```
$ printf '1,10,,100\n' | datamash -t, sum 4
100
```

5.4 Column Ranges

`datamash` accepts column ranges such as *1,2,3* and *1-3*.

Simulating input with multiple columns:

```
$ seq 100 | paste - - - -
1    2    3    4
5    6    7    8
9   10   11   12
13  14   15   16
17  18   19   20
...
```

The following are equivalent:

```
$ seq 100 | paste - - - - | datamash sum 1 sum 2 sum 3 sum 4
1225  1250   1275   1300

$ seq 100 | paste - - - - | datamash sum 1,2,3,4
1225  1250   1275   1300

$ seq 100 | paste - - - - | datamash sum 1-4
1225  1250   1275   1300

$ seq 100 | paste - - - - | datamash sum 1-3,4
```

```
1225  1250   1275   1300
```

Ranges can be used with multiple operations:

```
$ seq 100 | paste - - - - | datamash sum 1-4 mean 1-4
1225  1250   1275   1300   49   50   51   52
```

5.5 Reverse and Transpose

Transpose

Use `transpose` to swap rows and columns in a file:

```
$ cat input.txt
Sample   Year   Count
A        2014   1002
B        2013    990
C        2014   2030
D        2014    599

$ datamash transpose < input.txt
Sample  A       B       C       D
Year    2014    2013    2014    2014
Count   1002    990     2030    599
```

By default, `transpose` verifies the input has the same number of fields in each line, and fails with an error otherwise:

```
$ cat input.txt
Sample   Year   Count
A        2014   1002
B        2013
C        2014   2030
D        2014    599


$ datamash transpose < input1.txt
datamash: transpose input error: line 3 has 2 fields (previous lines had 3);
see --help to disable strict mode
```

Use `--no-strict` to allow missing values:

```
$ datamash --no-strict transpose < input1.txt
Sample  A       B        C        D
Year    2014    2013     2014     2014
Count   1002    N/A      2030     599
```

Use `--filler` to set the missing-field filler value:

```
$ datamash --no-strict --filler XYZ transpose < input1.txt
Sample  A       B        C        D
Year    2014    2013     2014     2014
Count   1002    XYZ      2030     599
```

Reverse

Use `reverse` to reverse the fields order in a file:

```
$ cat input.txt
Sample   Year   Count
A        2014   1002
B        2013    990
C        2014   2030
D        2014    599

$ datamash reverse < input.txt
Count   Year    Sample
1002    2014    A
990     2013    B
2030    2014    C
599     2014    D
```

By default, reverse verifies the input has the same number of fields in each line, and fails with an error otherwise. Use `--no-strict` to disable this behavior (see section above for an example).

Combining Reverse and Transpose

Reverse and Transpose can be combined to achieve various manipulations. (reminder: tac (`https://www.gnu.org/software/coreutils/tac`) can be used to reverse lines in a file):

```
$ cat input.txt
A       1       xx
B       2       yy
C       3       zz


$ tac input.txt
C       3       zz
B       2       yy
A       1       xx


$ tac input.txt | datamash reverse
zz      3       C
yy      2       B
xx      1       A


$ cat input.txt | datamash reverse | datamash transpose
xx      yy      zz
1       2       3
A       B       C

$ tac input.txt | datamash reverse | datamash transpose
```

```
zz      yy      xx
3       2       1
C       B       A
```

5.6 Groupby on /etc/passwd

`datamash` with the `groupby` operation mode can be used to aggregate information.

Using this simulated `/etc/passwd` file as input:

```
$ cat passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
mysql:x:115:124:MySQL Server,,,:/var/lib/mysql:/bin/false
sshd:x:116:65534::/var/run/sshd:/usr/sbin/nologin
guest:x:118:125:Guest,,,:/tmp/guest-home.phc17z:/bin/bash
gordon:x:1004:1000:Assaf Gordon,,,,:/home/gordon:/bin/bash
charles:x:1005:1000:Charles,,,,:/home/charles:/bin/bash
alice:x:1006:1000:Alice,,,,:/home/alice:/bin/bash
bob:x:1007:1000:Bob,,,,:/home/bob:/bin/bash
postgres:x:119:126:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
rabbitmq:x:125:138:RabbitMQ messaging server,,,:/var/lib/rabbitmq:/bin/false
redis:x:126:140:redis server,,,:/var/lib/redis:/bin/false
postfix:x:127:141::/var/spool/postfix:/bin/false
```

Parameter `-t` is used to indicate the field separator *:* (instead of the default *tab*).

Aggregate (`groupby`) login shells (column 7) and `count` how many users use each:

```
$ datamash -t: --sort groupby 7 count 7 < passwd
/bin/bash:7
/bin/false:4
/bin/sync:1
/usr/sbin/nologin:14
```

Aggregate (`groupby`) login shells (column 7) and print comma-separated list of users (column 1) for each shell (`collapse`):

```
$ cat passwd | datamash -t: --sort groupby 7 collapse 1
/bin/bash:root,guest,gordon,charles,alice,bob,postgres
```

```
/bin/false:mysql,rabbitmq,redis,postfix
/bin/sync:sync
/usr/sbin/nologin:daemon,bin,sys,games,man,lp,mail,news,uucp,proxy ,www-data,backup,li
```

Aggregate unix-groups (column 4) and print comma-separated list of users (column 1) for in each group:

```
$ datamash -t: --sort groupby 4 collapse 1 < /etc/passwd
0:root
1:daemon
10:uucp
1000:gordon,charles,alice,bob
12:man
124:mysql
125:guest
126:postgres
13:proxy
138:rabbitmq
140:redis
141:postfix
2:bin
3:sys
33:www-data
34:backup
38:list
60:games
65534:sync,sshd
7:lp
8:mail
9:news
```

5.7 Check - checking tabular structure

`datamash check` validates the tabular structure of a file, ensuring all lines have the same number of fields. `check` is meant to be used in scripting and automation pipelines, as it will terminate with non-zero exit code if the file is not well structured, while also printing detailed context information about the offending lines:

```
$ cat good.txt
A    1    ww
B    2    xx
C    3    yy
D    4    zz


$ cat bad.txt
A    1    ww
B    2    xx
C    3
D    4    zz
```

```
$ datamash check < good.txt && echo ok || echo fail
4 lines, 3 fields
ok


$ datamash check < bad.txt && echo ok || echo fail
line 2 (3 fields):
  B  2 xx
line 3 (2 fields):
  C  3
datamash: check failed: line 3 has 2 fields (previous line had 3)
fail
```

5.7.1 Expected number of lines/fields

`check` accepts optional *lines* and *fields* and will return failure if the input does not have the requested number of lines/fields.

The syntax is:

```
datamash check [N lines] [N fields]
```

Usage examples:

```
$ cat file.txt
A    1    ww
B    2    xx
C    3    yy
D    4    zz

$ datamash check 4 lines < file.txt && echo ok
4 lines, 3 fields
ok

$ datamash check 3 fields < file.txt && echo ok
4 lines, 3 fields
ok

$ datamash check 4 lines 3 fields < file.txt && echo ok
4 lines, 3 fields
ok

$ datamash check 7 fields < file.txt && echo ok
line 1 (3 fields):
  A    1    ww
```

```
datamash: check failed: line 1 has 3 fields (expecting 22)

$ datamash check 10 lines < file.txt && echo ok
datamash: check failed: input had 4 lines (expecting 10)
```

For convenience, *line*,*row*,*rows* can be used instead of *lines*; *field*,*columns*,*column*,*col* can be used instead of *fields*. The following are all equivalent:

```
datamash check 4 lines 10 fields < file.txt
datamash check 4 rows  10 columns < file.txt
datamash check 10 col 4 row < file.txt
```

5.7.2 checks in automation scripts

In pipeline/automation context, it is often beneficial to validate files as early as possible (immediately after file is created, as in fail-fast methodology (`https://en.wikipedia.org/wiki/Fail-fast`)). A typical usage in a shell script would be:

```
#!/bin/sh

die()
{
    base=$(basename "$0")
    echo "$base: error: $@" >&2
    exit 1
}

custom pipeline-or-program > output.txt \
    || die "program failed"

datamash check < output.txt \
    || die "'output.txt' has invalid structure (missing fields)"
```

If the generated `output.txt` file has invalid structure (i.e. missing fields), `datamash` will print the `stderr` enough details to help in troubleshooting (line numbers and offending line's content).

5.8 Crosstab - Cross-Tabulation (pivot-tables)

Cross-tabulation compares the relationship between two fields. Given the following input file:

```
$ cat input.txt
a    x    3
a    y    7
b    x    21
a    x    40
```

Show cross-tabulation between the first field (a/b) and the second field (x/y) - counting how many times each pair appears (note: sorting is required):

```
$ datamash -s crosstab 1,2 < input.txt
     x    y
```

```
a    2    1
b    1    N/A
```

The default operation is `count` - in the above example, *a* and *x* appear twice in the input file, while *b* and *y* never appear together.

An optional grouping operation can be used instead of counting.

For each pair, `sum` the values in the third column:

```
$ datamash -s crosstab 1,2 sum 3 < input.txt
     x    y
a    43   7
b    21   N/A
```

For each pair, list all `unique` values in the third column:

```
$ datamash -s crosstab 1,2 unique 3 < input.txt
     x    y
a    3,40 7
b    21   N/A
```

Note that using `--header-out` with crosstab prints a line showing how to interpret the rows and columns, and what operation was used.

```
$ datamash -s --header-in --header-out crosstab 1,2 < input.txt
GroupBy(a) GroupBy(x) count(a)
     x    y
a    1    1
b    1    N/A
```

5.9 Rounding numbers

The following demonstrate the different rounding operations:

```
$ ( echo X ; seq -1.25 0.25 1.25 ) \
      | datamash --full -H round 1 ceil 1 floor 1 trunc 1 frac 1

  X     round(X)  ceil(X)  floor(X)  trunc(X)   frac(X)
-1.25   -1        -1       -2        -1         -0.25
-1.00   -1        -1       -1        -1          0
-0.75   -1         0       -1         0         -0.75
-0.50   -1         0       -1         0         -0.5
-0.25    0         0       -1         0         -0.25
 0.00    0         0        0         0          0
 0.25    0         1        0         0          0.25
 0.50    1         1        0         0          0.5
 0.75    1         1        0         0          0.75
 1.00    1         1        1         1          0
 1.25    1         2        1         1          0.25
```

5.10 Binning numbers

Bin input values into buckets of size 5:

```
$ ( echo X ; seq -10 2.5 10 ) \
```

```
      | datamash -H --full bin:5 1
     X  bin(X)
-10.0    -10
 -7.5    -10
 -5.0     -5
 -2.5     -5
  0.0      0
  2.5      0
  5.0      5
  7.5      5
 10.0     10
```

5.11 Binning strings

Hash any string input value into a numeric integer. A typical usage would be to split an input file into N chunks, ensuring that all values of a certain key will be stored in the same chunk:

```
$ cat input.txt
PatientA   10
PatientB   11
PatientC   12
PatientA   14
PatientC   15
```

Each patient ID is hashed into a bin between 0 and 9 and printed in the last field:

```
$ datamash --full strbin 1 < input.txt
PatientA   10    5
PatientB   11    6
PatientC   12    7
PatientA   14    5
PatientC   15    7
```

Splitting the input into chunks can be done with awk:

```
$ cat input.txt | datamash --full strbin 1 \
    | awk '{print > $NF ".txt"}'
```

5.12 Extracting numeric values - using getnum

The `getnum` operation extracts a numeric value from the field:

```
$ echo zoom-123.45xyz | datamash getnum 1
123.45
```

`getnum` accepts an optional single-letter *TYPE* option:

`getnum:n` natural numbers (positive integers, including zero)

`getnum:i` integers

`getnum:d` decimal point numbers

`getnum:p` positive decimal point numbers (this is the default)

`getnum:h` hex numbers

`getnum:o` octal numbers

Examples:

```
$ echo zoom-123.45xyz | datamash getnum 1
123.45

$ echo zoom-123.45xyz | datamash getnum:n 1
123

$ echo zoom-123.45xyz | datamash getnum:i 1
-123

$ echo zoom-123.45xyz | datamash getnum:d 1
123.45

$ echo zoom-123.45xyz | datamash getnum:p 1
-123.45

# Hex 0x123 = 291 Decimal
$ echo zoom-123.45xyz | datamash getnum:h 1
291

# Octal 0123 = 83 Decimal
$ echo zoom-123.45xyz | datamash getnum:o 1
83
```

6 Reporting bugs

To report bugs, suggest enhancements or otherwise discuss GNU Datamash, please send electronic mail to `bug-datamash@gnu.org`.

For bug reports, please include enough information for the maintainers to reproduce the problem. Generally speaking, that means:

- The version numbers of Datamash (which you can find by running '`datamash --version`') and any other program(s) or manual(s) involved.
- Hardware and operating system names and versions.
- The contents of any input files necessary to reproduce the bug.
- The expected behavior and/or output.
- A description of the problem and samples of any erroneous output.
- Options you gave to `configure` other than specifying installation directories.
- Anything else that you think would be helpful.

When in doubt whether something is needed or not, include it. It's better to include too much than to leave out something important.

Patches are welcome; if possible, please make them with '`diff -u`' (see Section "Overview" in *Comparing and Merging Files*) and include `ChangeLog` entries (see Section "Change Log" in *The GNU Emacs Manual*). Please follow the existing coding style.

Appendix A GNU Free Documentation License

Version 1.3, 3 November 2008

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document *free* in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released

under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

   If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

   If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

   If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

   It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

   You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

   A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any,

be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their

titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

   You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

   The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

   In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

   You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

   You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

   A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

   If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

   Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

   If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

   You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

   However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

   Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

   Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

    The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See `https://www.gnu.org/licenses/`.

    Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

    "Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

    "CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

    "Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

    An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

    The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (C)  year  your name.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.3
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover
Texts.  A copy of the license is included in the section entitled ``GNU
Free Documentation License''.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with. . . Texts." line with this:

```
with the Invariant Sections being list their titles, with
the Front-Cover Texts being list, and with the Back-Cover Texts
being list.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Concept index

G

H

I

L

M

N

O

P

Q

R

S

T

U

W