

# Productivity tools II: Datamash

UZH, July 12th, 2018

# Datamash GNU



GNU Operating System

Sponsored by the *Free Software Foundation*

JOIN THE FSF

Free Software Supporter

email address

Sign up

[ABOUT GNU](#) [PHILOSOPHY](#) [LICENSES](#) [EDUCATION](#) [SOFTWARE](#) [DOCUMENTATION](#) [HELP GNU](#)

## GNU datamash

GNU datamash is a command-line program which performs basic numeric, textual and statistical operations on input textual data files.

Examples:

calculate the sum and mean of values 1 to 10:

```
$ seq 10 | datamash sum 1 mean 1
55 5.5
```

group text file by one column and calculate  
mean and sample standard deviation on another,  
with automatic sorting and header line processing:

```
$ datamash --sort --headers groupby 2 mean 3 sstdev 3 < scores_h.txt
GroupBy(Major)  mean(Score)  sstdev(Score)
Arts            68.94        10.42
...
```

file validation for pipeline automation and troubleshooting:

```
$ datamash check < snp147Common.txt && echo ok || echo fail
15189820 lines, 26 fields
ok
```

<https://www.gnu.org/software/datamash/>

<https://www.gnu.org/software/datamash/download/>

# Performs calculations: *sum, count, mean, sstdev...*

General format (datamash --help):

```
datamash [FUNCTION] [COLUMN] ... [FUNCTION2] [COLUMN2]
```

FUNCTION is the operation to be performed and COLUMN is the field you want to perform the operation on.

**NOTE:** The default delimiter of datamash is tabular, but it can be customized

<https://www.gnu.org/software/datamash/manual/datamash.html>

# Usage: Input files and functions

## Examples:

```
datamash sum 1 sum 2 < inputfile          #option 1, input file with "<"  
cat inputfile | datamash sum 1 sum 2      #option 2, input file with "cat"
```

Option #1 is **faster** and more straightforward.

# Examples: awk vs datamash

Simple sum operation:

Awk:

```
$ seq 10 | awk '{sum+=$1}END{print sum}'
```

55

---

Datamash:

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

# much shorter!

55

# Examples: datamash

## Sum and Range operations:

```
$ seq 100 | paste - - - - | datamash sum 1 sum 2 sum 3 sum 4
```

```
$ seq 100 | paste - - - - | datamash sum 1,2,3,4
```

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

## Combined operations: sum, range and mean

```
$ seq 100 | paste - - - - | datamash sum 1-4 mean 1-4
```

# Many different functions

Transpose, collapse, GroupBy, Collapse, Reverse...

```
$ datamash -t: --sort groupby 7 collapse 1 < /etc/passwd
```

```
/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,list,sshd
```

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

# Check the documentation for more!

## GNU Datamash 1.3

### Table of Contents

- 1 Overview
- 2 Invoking datamash
- 3 Available operations in datamash
- 4 Statistical Operations
  - Equivalent R functions
- 5 Usage Examples
  - 5.1 Summary Statistics
  - 5.2 Header Lines and Column Names
    - Output Header Lines
    - Skipping Input Header Lines
    - Using Header Lines
    - Column Names
  - 5.3 Field Delimiters
  - 5.4 Column Ranges
  - 5.5 Reverse and Transpose
    - Transpose
    - Reverse
    - Combining Reverse and Transpose
  - 5.6 Groupby on /etc/passwd
  - 5.7 Check - checking tabular structure
    - 5.7.1 Expected number of lines/fields
    - 5.7.2 checks in automation scripts
  - 5.8 Crosstab - Cross-Tabulation (pivot-tables)
  - 5.9 Rounding numbers
  - 5.10 Binning numbers
  - 5.11 Binning strings
- 6 Reporting bugs
- Appendix A GNU Free Documentation License
- Concept index

<https://www.gnu.org/software/datamash/manual/datamash.html>

Next: Overview, Up: (dir) [Contents][Index]

## Datamash

This manual is for GNU Datamash (version 1.3, 25 January 2018), which provides command-line computations on input files.

- |                           |                                     |
|---------------------------|-------------------------------------|
| • Overview:               | General purpose and information.    |
| • Invoking datamash:      | How to run datamash.                |
| • Available Operations:   | Available operations in datamash.   |
| • Statistical Operations: | Statistical operations in datamash. |



# Hands-on session!

- 1) **Go to:** [https://github.com/carlaibc/URPP\\_tutorials/](https://github.com/carlaibc/URPP_tutorials/)
- 2) Follow the instructions!
- 3) For more go to:  
<https://www.gnu.org/software/datamash/manual/datamash.html>

**NOTE:** Use **column -ts**, for easier visualization of the tables in the terminal  
(example is for a **comma-separated** file)

# Productivity tools II: Terminal Tricks

UZH, July 12th, 2018

# Edit your .bashrc

The best thing in the World are **aliases**!

For example, instead of:

```
$ sed 's/,/\t/g' infile
```

 ← Modifying a comma-separated file for quick visualization can be tiresome. Make an alias for it and add it to your **.bashrc**!

```
# ~/.bashrc:
```

```
#my aliases
```

```
alias subcomma='sed 's/,/\t/g'' #comma-separated to tabular-separated file
```

```
$ source ~/.bashrc
```

 ← Activate your new configuration

You can make aliases of **everything**!

# Useful aliases

```
# ~/.bashrc:
```

```
#my aliases
```

```
alias h='head'
```

```
alias ..='cd ..'
```

```
alias rm='rm -i'
```

```
alias untar='tar xvzf'
```

```
#shorter!
```

```
# no more cd!
```

```
#do you really want to delete that file?
```

```
#difficult to remember!
```

```
$ source ~/.bashrc
```

← remember to activate your new configuration

# Useful bioinformatic one-liners

1) Count how many sequences you have in a FASTA file:

```
$ grep -e '^>' file.fasta | wc -l
```

2) Reverse complement a DNA sequence:

```
$ echo ATG | tr "ATGcatgc" "TACGtacg" | rev
```

3) Count the records in a gzipped FASTQ file:

```
$ zcat seqs.fastq.gz | echo $((`wc -l`/4))
```

