

# Contents

1	<b>1 Guide for trident v1.1.7.0</b>	<b>1</b>
2	1.1 The trident CLI	1
3	1.1.1 Handling data with trident	2
4	1.1.2 Notes on duplicates	3
5	1.2 Package creation and manipulation commands	4
6	1.2.1 Init command	4
7	1.2.2 Fetch command	5
8	1.2.3 Forge command	6
9	1.2.4 Genoconvert command	11
10	1.2.5 Update command	13
11	1.3 Inspection commands	14
12	1.3.1 List command	14
13	1.3.2 Summarise command	16
14	1.3.3 Survey command	16
15	1.3.4 Validate command	17
16		

## 1 Guide for trident v1.1.7.0

### 1.1 The trident CLI

Trident is a command line software tool structured in multiple subcommands. If you installed it properly you can call it on the command line by typing `trident`. This will show an overview of the general options and all subcommands, which are explained in detail below.

```
Usage: trident [--version] [--logMode ARG] [--errLength ARG] (COMMAND | COMMAND)
trident is a management and analysis tool for Poseidon packages. Report issues
here: https://github.com/poseidon-framework/poseidon-hs/issues
```

#### Available options:

<code>-h,--help</code>	Show this help text
<code>--version</code>	Show version number
<code>--logMode ARG</code>	How information should be reported: NoLog, SimpleLog, DefaultLog, ServerLog or VerboseLog (default: DefaultLog)
<code>--errLength ARG</code>	After how many characters should a potential error message be truncated. "Inf" for no truncation. (default: CharCount 1500)

#### Package creation and manipulation commands:

<code>init</code>	Create a new Poseidon package from genotype data
<code>fetch</code>	Download data from a remote Poseidon repository
<code>forge</code>	Select packages, groups or individuals and create a new Poseidon package from them
<code>genoconvert</code>	Convert the genotype data in a Poseidon package to a different file format

```

43     update                Update POSEIDON.yml files automatically
44
45 Inspection commands:
46     list                  List packages, groups or individuals from local or
47                           remote Poseidon repositories
48     summarise             Get an overview over the content of one or multiple
49                           Poseidon packages
50     summarize             Synonym for summarise
51     survey                Survey the degree of context information completeness
52                           for Poseidon packages
53     validate              Check one or multiple Poseidon packages for
54                           structural correctness
55
56 For all subcommands the general argument --logMode defines how trident reports messages (to stderr) on the
57 command line:
58
59     • NoLog: Hides all messages.
60     • SimpleLog: Plain and simple output to stderr.
61     • DefaultLog: Adds severity indicators before each message. (default setting)
62     • ServerLog: Additionally adds timestamps before each message.
63     • VerboseLog: Shows not just messages on the log levels Info, Warning and Error like the other modes, but
64       also on the more verbose level Debug. Use this for debugging.

```

### 63 1.1.1 Handling data with trident

64 Trident allows to work directly with genotype data (see -p below), but its optimized for the interaction with  
65 [Poseidon packages](#), which wrap and contextualize the data. Most trident subcommands therefore have a central  
66 parameter, called --baseDir or simply -d to specify one or more base directories to look for packages. For example,  
67 if all Poseidon packages live inside a repository at /path/to/poseidon/packages you would simply say trident  
68 <subcommand> -d /path/to/poseidon/dirs/ and trident would automatically search all subdirectories inside  
69 of the repository for valid Poseidon packages (as identified by valid POSEIDON.yml files).

70 You can arrange a poseidon repository in a hierarchical way. For example:

```

71 /path/to/poseidon/packages
72     /modern
73         /2019_poseidon_package1
74         /2019_poseidon_package2
75     /ancient
76         /...
77         /...
78     /Reference_Genomes
79         /...
80         /...

```

81 You can use this structure to select only the level of packages you're interested in, even individual ones, and you  
82 can make use of the fact that -d can be given multiple times.

83 Being able to specify one or multiple repositories is often not enough, as you may have your own data to  
84 co-analyse with the main repository. This is easy to do, as you simply need to provide your own genotype data as

85 yet another Poseidon package to be added to your `trident` command. For example, let's say you have genotype  
86 data in EIGENSTRAT format (`trident` supports EIGENSTRAT and PLINK as formats.):

```
87 ~/my_project/my_project.geno
88 ~/my_project/my_project.snp
89 ~/my_project/my_project.ind
```

90 then you can make that to a skeleton Poseidon package with the `init` command. You can also do it manually by  
91 simply adding a POSEIDON.yml file, with for example the following content:

```
92 poseidonVersion: 2.5.0
93 title: My_awesome_project
94 description: Unpublished genetic data from my awesome project
95 contributor:
96   - name: Stephan Schiffels
97     email: schiffels@institute.org
98 packageVersion: 0.1.0
99 lastModified: 2020-10-07
100 genotypeData:
101   format: EIGENSTRAT
102   genoFile: my_project.geno
103   snpFile: my_project.snp
104   indFile: my_project.ind
105   jannoFile: my_project.janno
106   bibFile: sources.bib
```

107 Two remarks: 1) all file paths are considered *relative* to the directory in which POSEIDON.yml resides. Here we  
108 assume that you put this file into the same directory as the three genotype files. 2) Besides the genotype data  
109 files there are two (technically optional) files referenced by this example POSEIDON.yml file: `sources.bib` and  
110 `my_project.janno`. Of course you can add them manually - `init` automatically creates empty dummy versions.

111 Once you have set up your own “Poseidon” package (which is really only a skeleton so far), you can add it to  
112 your `trident` analysis, by simply adding your project directory to the command using `-d`, for example:

```
113 trident list -d /path/to/poseidon/packages/modern \
114   -d /path/to/poseidon/packages/ReferenceGenomes
115   -d ~/my_project --packages
```

### 116 1.1.2 Notes on duplicates

- 117 • If multiple packages in a package repository share the same `title`, then `trident` will try to select the one  
118 with the highest version number. If this is not sufficient to resolve the conflict, `trident` will stop.
- 119 • Individual/sample names (`Poseidon_IDs`) within one package have to be unique, or `trident` will stop.
- 120 • We generally also discourage ID duplicates across packages in package repositories, but `trident` will generally  
121 continue with them after printing a warning. This does not apply for `validate`, by default (you can  
122 change this behaviour with `--ignoreDuplicates`), and `forge`. `forge` offers a special mechanism to resolve  
123 duplicates within its selection language (see below).

## 1.2 Package creation and manipulation commands

### 1.2.1 Init command

`init` creates a new, valid Poseidon package from genotype data files. It adds a valid `POSEIDON.yml` file, a dummy `.janno` file for context information and an empty `.bib` file for literature references.

[Click here for command line details](#)

```
Usage: trident init ((-p|--genoOne ARG) | --inFormat ARG --genoFile ARG
                    --snpFile ARG --indFile ARG) [--snpSet ARG]
                    (-o|--outPackagePath ARG) [-n|--outPackageName ARG]
                    [--minimal]
```

Create a new Poseidon package from genotype data

Available options:

<code>-h,--help</code>	Show this help text
<code>-p,--genoOne ARG</code>	one of the input genotype data files. Expects <code>.bed</code> or <code>.bim</code> or <code>.fam</code> for PLINK and <code>.geno</code> or <code>.snp</code> or <code>.ind</code> for EIGENSTRAT. The other files must be in the same directory and must have the same base name
<code>--inFormat ARG</code>	the format of the input genotype data: EIGENSTRAT or PLINK (only necessary for data input with <code>--genoFile</code> + <code>--snpFile</code> + <code>--indFile</code> )
<code>--genoFile ARG</code>	the input geno file path
<code>--snpFile ARG</code>	the input snp file path
<code>--indFile ARG</code>	the input ind file path
<code>--snpSet ARG</code>	the snpSet of the package: 1240K, HumanOrigins or Other. (only relevant for data input with <code>-p --genoOne</code> or <code>--genoFile</code> + <code>--snpFile</code> + <code>--indFile</code> , because the packages in a <code>-d --baseDir</code> already have this information in their respective <code>POSEIDON.yml</code> files) Default: Other
<code>-o,--outPackagePath ARG</code>	the output package directory path
<code>-n,--outPackageName ARG</code>	the output package name - this is optional: If no name is provided, then the package name defaults to the basename of the (mandatory) <code>--outPackagePath</code> argument
<code>--minimal</code>	should only a minimal output package be created?

The command

```
trident init \
  --inFormat EIGENSTRAT/PLINK \
  --genoFile path/to/geno_file \
  --snpFile path/to/snp_file \
  --indFile path/to/ind_file \
  --snpSet 1240K|HumanOrigins|Other \
  -o path/to/new_package_name
```

requires the format (`--inFormat`) of your input data (either EIGENSTRAT or PLINK), the paths to the respective files (`--genoFile`, `--snpFile`, `--indFile`), and optionally the “shape” of these files (`--snpSet`), so if they cover the 1240K, the HumanOrigins or an Other SNP set. A simpler interface added in trident 0.29.0 is available with `-p (+ --snpSet)`.

	EIGENSTRAT	PLINK
genoFile	.geno	.bed
snpFile	.snp	.bim
indFile	.ind	.fam

The output package of `init` is created as a new directory `-o`, which should not already exist, and gets the package `title` corresponding to the basename of `-o`. You can also set the title explicitly with `-n`. The `--minimal` flag causes `init` to create a minimal package with a very basic POSEIDON.yml and no .bib and .janno files.

### 1.2.2 Fetch command

`fetch` allows to download Poseidon packages from a remote Poseidon server. Read more about this repository [here](#).

Click here for command line details

```
Usage: trident fetch (-d|--baseDir DIR)
        (--downloadAll |
        (--fetchFile ARG | (-f|--fetchString ARG)))
        [--remoteURL ARG] [-u|--upgrade]
```

Download data from a remote Poseidon repository

Available options:

<code>-h,--help</code>	Show this help text
<code>-d,--baseDir DIR</code>	a base directory to search for Poseidon Packages (could be a Poseidon repository)
<code>--downloadAll</code>	download all packages the server is offering
<code>--fetchFile ARG</code>	A file with a list of packages. Works just as <code>-f</code> , but multiple values can also be separated by newline, not just by comma. <code>-f</code> and <code>--fetchFile</code> can be combined.
<code>-f,--fetchString ARG</code>	List of packages to be downloaded from the remote server. Package names should be wrapped in asterisks: <code>*package_title*</code> . You can combine multiple values with comma, so for example: <code>"*package_1*, *package_2*, *package_3*"</code> . <code>fetchString</code> uses the same parser as <code>forgeString</code> , but does not allow excludes. If groups or individuals are specified, then packages which include these groups or individuals are included in the download.
<code>--remoteURL ARG</code>	URL of the remote Poseidon server (default: "https://c107-224.cloud.gwdg.de")
<code>-u,--upgrade</code>	overwrite outdated local package versions

204 It works with

```
205 trident fetch -d ... -d ... \
206 -f "*package_title_1*,*package_title_2*,*package_title_3*,group_name,<Individual1>"
```

207 and the entities you want to download must be listed either in a simple string of comma-separated values, which  
 208 can be passed via `-f/--fetchString`, or in a text file (`--fetchFile`). Entities are then combined from these  
 209 sources.

210 Entities are specified using a special syntax (see also the documentation of **forge** below): Package titles are  
 211 wrapped in asterisks: *package\_title*, group names are spelled as is, and individual names are wrapped in angular  
 212 brackets, like `<Individual1>`. Fetch will figure out which packages need to be downloaded to include all specified  
 213 entities. `--downloadAll`, which can be given instead of `-f` and `--fetchFile`, causes fetch to download all  
 214 packages from the server. The downloaded packages are added in the first (!) `-d` directory (which gets created  
 215 if it doesn't exist), but downloads are only performed if the respective packages are not already present in an  
 216 up-to-date version in any of the `-d` dirs.

217 Note that **trident fetch** makes most sense in combination with **trident list --remote**: First one can inspect  
 218 what is available on the server, then one can create a custom fetch command.

219 **fetch** also has the optional arguments `--remote https://...` to name an alternative poseidon server. The  
 220 default points to the **DAG server**.

221 To overwrite outdated package versions with **fetch**, the `-u/--upgrade` flag has to be set. Note that many file  
 222 systems do not offer a way to recover overwritten files. So be careful with this switch.

### 223 1.2.3 Forge command

224 **forge** creates new Poseidon packages by extracting and merging packages, populations and individuals from  
 225 your Poseidon repositories.

226 Click here for command line details

```
227 Usage: trident forge ((-d|--baseDir DIR) |
228                  ((-p|--genoOne ARG) | --inFormat ARG --genoFile ARG
229                  --snpFile ARG --indFile ARG) [--snpSet ARG])
230                  [--forgeFile ARG | (-f|--forgeString ARG)]
231                  [--selectSnps ARG] [--intersect] [--outFormat ARG]
232                  [--minimal] [--onlyGeno] (-o|--outPackagePath ARG)
233                  [-n|--outPackageName ARG] [--no-extract]
```

234 Select packages, groups or individuals and create a new Poseidon package from  
 235 them

236

237 Available options:

238 <code>-h,--help</code>	Show this help text
239 <code>-d,--baseDir DIR</code>	a base directory to search for Poseidon Packages (could be a Poseidon repository)
240	
241 <code>-p,--genoOne ARG</code>	one of the input genotype data files. Expects .bed or 242 .bim or .fam for PLINK and .geno or .snp or .ind for 243 EIGENSTRAT. The other files must be in the same 244 directory and must have the same base name

245 --inFormat ARG the format of the input genotype data: EIGENSTRAT or  
246 PLINK (only necessary for data input with --genoFile  
247 + --snpFile + --indFile)  
248 --genoFile ARG the input geno file path  
249 --snpFile ARG the input snp file path  
250 --indFile ARG the input ind file path  
251 --snpSet ARG the snpSet of the package: 1240K, HumanOrigins or  
252 Other. (only relevant for data input with  
253 -p|--genoOne or --genoFile + --snpFile + --indFile,  
254 because the packages in a -d|--baseDir already have  
255 this information in their respective POSEIDON.yml  
256 files) Default: Other  
257 --forgeFile ARG A file with a list of packages, groups or individual  
258 samples. Works just as -f, but multiple values can  
259 also be separated by newline, not just by comma.  
260 Empty lines are ignored and comments start with "#",  
261 so everything after "#" is ignored in one line.  
262 Multiple instances of -f and --forgeFile can be  
263 given. They will be evaluated according to their  
264 input order on the command line.  
265 -f,--forgeString ARG List of packages, groups or individual samples to be  
266 combined in the output package. Packages follow the  
267 syntax \*package\_title\*, populations/groups are simply  
268 group\_id and individuals <individual\_id>. You can  
269 combine multiple values with comma, so for example:  
270 "\*package\_1\*, <individual\_1>, <individual\_2>,  
271 group\_1". Duplicates are treated as one entry.  
272 Negative selection is possible by prepending "-" to  
273 the entity you want to exclude (e.g. "\*package\_1\*,  
274 -<individual\_1>, -group\_1"). forge will apply  
275 excludes and includes in order. If the first entity  
276 is negative, then forge will assume you want to merge  
277 all individuals in the packages found in the baseDirs  
278 (except the ones explicitly excluded) before the  
279 exclude entities are applied. An empty forgeString  
280 (and no --forgeFile) will therefore merge all  
281 available individuals. If there are individuals in  
282 your input packages with equal individual id, but  
283 different main group or source package, they can be  
284 specified with the special syntax  
285 "<package:group:individual>".  
286 --selectSnps ARG To extract specific SNPs during this forge operation,  
287 provide a Snp file. Can be either Eigenstrat (file  
288 ending must be '.snp') or Plink (file ending must be  
289 '.bim'). When this option is set, the output package

will have exactly the SNPs listed in this file. Any SNP not listed in the file will be excluded. If option '--intersect' is also set, only the SNPs overlapping between the SNP file and the forged packages are output.

--intersect Whether to output the intersection of the genotype files to be forged. The default (if this option is not set) is to output the union of all SNPs, with genotypes defined as missing in those packages which do not have a SNP that is present in another package. With this option set, the forged dataset will typically have fewer SNPs, but less missingness.

--outFormat ARG the format of the output genotype data: EIGENSTRAT or PLINK. Default: PLINK

--minimal should only a minimal output package be created?

--onlyGeno should only the resulting genotype data be returned? This means the output will not be a Poseidon package

-o,--outPackagePath ARG the output package directory path

-n,--outPackageName ARG the output package name - this is optional: If no name is provided, then the package name defaults to the basename of the (mandatory) --outPackagePath argument

--no-extract Skip the selection step in forge. This will result in outputting all individuals in the relevant packages, and hence a superset of the requested individuals/groups. It may result in better performance in cases where one wants to forge entire packages or almost entire packages. Note that this will also ignore any ordering in the output groups/individuals. With this option active, individuals from the relevant packages will just be written in the order that they appear in the original packages.

forge can be used with

```
trident forge -d ... -d ... \
  -f "*package_name*, group_id, <individual_id>" \
  -o path/to/new_package_name
```

where the entities (packages, groups/populations, individuals/samples) you want in the output package can be denoted either as a string on the command line (-f/--forgeString), or in an input text file (--forgeFile). See the section below for the syntax of this selection language. Do not forget to wrap the --forgeString query in quotes.

Including one or multiple Poseidon packages with -d is not the only way to include data for a forge operation. It is also possible to consider unpackaged genotype data directly with -p (+ --snpSet) or --inFormat + --genoFile + --snpFile + --indFile (+ --snpSet). This makes the following example possible, where we



334 merge data from one Poseidon package and two genotype datasets to get a new EIGENSTRAT dataset.

```
335 trident forge \  
336 -d 2017_GonzalesFortesCurrentBiology \  
337 -p 2018_VeeramahPNAS/2018_VeeramahPNAS.fam \  
338 --inFormat PLINK \  
339 --genoFile 2017_HaberAJHG/2017_HaberAJHG.bed \  
340 --snpFile 2017_HaberAJHG/2017_HaberAJHG.bim \  
341 --indFile 2017_HaberAJHG/2017_HaberAJHG.fam \  
342 -f "<STR241.SG>,<ERS1790729.SG>,Iberia_HG.SG" \  
343 -o testpackage \  
344 --outFormat EIGENSTRAT \  
345 --onlyGeno
```

346 **1.2.3.1 The forge selection language** The text in `--forgeString` and `--forgeFile` are parsed as a  
347 domain specific query language that describes precisely which entities should be compiled in the output package  
348 of a given `forge` operation. The language has multiple syntactic elements and a specific evaluation logic.

349 In general a `--forgeString` query consists of multiple entities, separated by `,`. The main entities are Poseidon  
350 packages, groups/populations and individuals/samples:

- 351 • Each package title is surrounded by `*`: `*package*`. That means if you want all individuals of the Poseidon  
352 package 2019\_Jeong\_InnerEurasia in the output package you would add `*2019_Jeong_InnerEurasia*`  
353 to the query.
- 354 • Groups/populations are not specially marked: `group`. So to get all individuals of the group  
355 `Swiss_Roman_period`, you would simply add `Swiss_Roman_period`.
- 356 • Individuals/samples are surrounded by `<` and `>`: `<individual>`. ALA026 therefore becomes `<ALA026>`. A sec-  
357 ond way to denote individuals is with the more verbose and specific syntax `<package:group:individual>`.  
358 Such defined individuals take precedence over differently defined ones (so: directly with `<individual>` or  
359 as a subset of `*package*` or `group`). This allows to resolve duplication issues precisely – at least in cases  
360 where the duplicated individuals differ in source package or primary group.

361 In the `--forgeFile` each line is treated as a separate `forgeString`, empty lines are ignored and `#`s start comments.  
362 So this is a valid `forgeFile`:

```
363 # Packages  
364 *package1*, *package2*  
365  
366 # Groups and individuals from other packages beyond package1 and package2  
367 group1, <individual1>, group2, <individual2>, <individual3>  
368  
369 # group2 has two outlier individuals that should be ignored  
370 -<bad_individual1> # This one has very low coverage  
371 -<bad_individual2> # This one is from a different time period
```

372 By prepending `-` to the bad individuals, we can exclude them from the forged package. `forge` fig-  
373 ures out the final list of samples to include by executing all `forge`-entities in order. So an entity list  
374 `*PackageA*, -<Individual1>, GroupA` may result in a different outcome than `*PackageA*, GroupA, -<Individual1>`,  
375 depending on whether `<Individual1>` belongs to `GroupA` or not. If the `forge` entity list starts with a negative

entity, or if the entity list is empty, **forge** will implicitly assume you want to include all individuals in all packages found in the baseDirs (except the ones explicitly excluded, of course).

An empty forgeString will therefore merge all available individuals.

**1.2.3.2 Treatment of the .janno file while merging** **forge** merges and subsets .janno files along with the genotype data. If a package lacks a .janno file, then a basic one will be created internally based on the information in the genotype data, and used for the output. Missing columns across packages will be filled with **n/a**.

For merging two .janno files **A** and **B** the following rules apply regarding undefined, arbitrary additional columns:

- If **A** has an additional column which is not in **B** then empty cells in the rows imported from **B** are filled with **n/a**.
- If **A** and **B** share additional columns with identical column name, then they are treated as semantically identical units and merged accordingly.
- In the resulting .janno file, all additional columns from both **A** and **B** are sorted alphabetically and appended after the normal, specified variables.

The following example illustrates the described behaviour:

#### A.janno

Poseidon_ID	Group_Name	Genetic_Sex	AdditionalColumn1	AdditionalColumn2
XXX011	POP1	M	A	D
XXX012	POP2	F	B	E
XXX013	POP1	M	C	F

#### B.janno

Poseidon_ID	Group_Name	Genetic_Sex	AdditionalColumn3	AdditionalColumn2
YYY022	POP5	F	G	J
YYY023	POP5	F	H	K
YYY024	POP5	M	I	L

#### A.janno + B.janno

Poseidon_ID	Group_Name	Genetic_Sex	AdditionalColumn1	AdditionalColumn2	AdditionalColumn3
XXX011	POP1	M	A	D	n/a
XXX012	POP2	F	B	E	n/a
XXX013	POP1	M	C	F	n/a
YYY022	POP5	F	n/a	J	G
YYY023	POP5	F	n/a	K	H
YYY024	POP5	M	n/a	L	I

394 **1.2.3.3 Other options** Just as for `init` the output package of `forge` is created as a new directory `-o`. The  
395 title can also be explicitly defined with `-n`.

396 `--minimal` allows for the creation of a minimal output package without `.bib` and `.janno`. This is especially  
397 useful for data analysis pipelines, where only the genotype data is required. Even more basic output comes with  
398 `--onlyGeno`, which means that only the genotype data is returned without any Poseidon package.

399 `forge` has a an optional flag `--intersect`, that defines, if the genotype data from different packages should  
400 be merged with an **union** or an **intersect** operation. The default (if this option is not set) is to output the  
401 union of all SNPs, with genotypes defined as missing in samples from packages which do not have a SNP that is  
402 present in another package. With this option set, on the other hand, the forged dataset will typically have fewer  
403 SNPs, but less missingness.

404 `--intersect` also influences the automatic determination of the `snpSet` field in the `POSEIDON.yml` file for the  
405 resulting package. If the `snpSets` of all input packages are identical, then the resulting package will just inherit  
406 this configuration. Otherwise `forge` applies the following pairwise merging logic:

Input snpSet A	Input snpSet B	<code>--intersect</code>	Ouput snpSet
Other	*	*	Other
1240K	HumanOrigins	True	HumanOrigins
1240K	HumanOrigins	False	1240K

407 `--selectSnps` allows to provide `forge` with a SNP file in EIGENSTRAT (`.snp`) or PLINK (`.bim`) format to  
408 create a package with a specific selection. When this option is set, the output package will have exactly the  
409 SNPs listed in this file. Any SNP not listed in the file will be excluded. If `--intersect` is also set, only the  
410 SNPs overlapping between the SNP file and the forged packages are output.

411 Merging genotype data across different data sources and file formats is tricky. `forge` is more verbose about  
412 potential issues, if the `--logMode` flag is set to `VerboseLog`.

413 The `--onlyGeno` command specifies that only genotype data should be output, not an entire Poseidon package.

#### 414 1.2.4 Genoconvert command

415 `genoconvert` converts the genotype data in a Poseidon package to a different file format. The respective entries  
416 in the `POSEIDON.yml` file are changed accordingly.

417 [Click here for command line details](#)

```
418 Usage: trident genoconvert ((-d|--baseDir DIR) |
419                             ((-p|--genoOne ARG) | --inFormat ARG --genoFile ARG
420                             --snpFile ARG --indFile ARG) [--snpSet ARG])
421                             --outFormat ARG [--onlyGeno]
422                             [-o|--outPackagePath ARG] [--removeOld]
```

423 Convert the genotype data in a Poseidon package to a different file format

424 Available options:

```
425 -h,--help          Show this help text
426 -d,--baseDir DIR   a base directory to search for Poseidon Packages
```

```

428                                     (could be a Poseidon repository)
429  -p,--genoOne ARG                   one of the input genotype data files. Expects .bed or
430                                     .bim or .fam for PLINK and .geno or .snp or .ind for
431                                     EIGENSTRAT. The other files must be in the same
432                                     directory and must have the same base name
433  --inFormat ARG                     the format of the input genotype data: EIGENSTRAT or
434                                     PLINK (only necessary for data input with --genoFile
435                                     + --snpFile + --indFile)
436  --genoFile ARG                     the input geno file path
437  --snpFile ARG                      the input snp file path
438  --indFile ARG                      the input ind file path
439  --snpSet ARG                       the snpSet of the package: 1240K, HumanOrigins or
440                                     Other. (only relevant for data input with
441                                     -p|--genoOne or --genoFile + --snpFile + --indFile,
442                                     because the packages in a -d|--baseDir already have
443                                     this information in their respective POSEIDON.yml
444                                     files) Default: Other
445  --outFormat ARG                    the format of the output genotype data: EIGENSTRAT or
446                                     PLINK.
447  --onlyGeno                         should only the resulting genotype data be returned?
448                                     This means the output will not be a Poseidon package
449  -o,--outPackagePath ARG            the output package directory path - this is optional:
450                                     If no path is provided, then the output is written to
451                                     the directories where the input genotype data file
452                                     (.bed/.geno) is stored
453  --removeOld                       Remove the old genotype files when creating the new
454                                     ones
455
456  With the default setting
457
458  trident genoconvert -d ... -d ... --outFormat EIGENSTRAT|PLINK
459
460  all packages in -d will be converted to the desired --outFormat (either EIGENSTRAT or PLINK), if the data is
461  not already in this format. This includes updating the respective POSEIDON.yml files.
462
463  The “old” data is not deleted, but kept around. That means conversion can result in a package with both PLINK
464  and EIGENSTRAT data, but only one is linked in the POSEIDON.yml file, and that is what will be used by
465  trident. To delete the old data in the conversion you can add the --removeOld flag.
466
467  Instead of -d to change Poseidon packages, the -p (+ --snpSet) or --inFormat + --genoFile + --snpFile
468  + --indFile (+ --snpSet) allow to directly convert genotype data that is not wrapped in a Poseidon package
469  and store it to a directory given in -o. See this example:
470
471  trident genoconvert \
472    -p 2018_Mittnik_Baltic/Mittnik_Baltic.bed \
473    --outFormat EIGENSTRAT
474    -o my_directory

```

### 469 1.2.5 Update command

470 **update** automatically harmonizes POSEIDON.yml files of one or multiple packages if the packages were changed.  
471 This is not an automatic update from one Poseidon version to the next!

472 Click here for command line details

```
473 Usage: trident update (-d|--baseDir DIR) [--poseidonVersion ARG]
474                [--ignorePoseidonVersion] [--versionComponent ARG]
475                [--noChecksumUpdate] [--newContributors ARG]
476                [--logText ARG] [--force]
```

477 Update POSEIDON.yml files automatically

478  
479 Available options:

480	-h,--help	Show this help text
481	-d,--baseDir DIR	a base directory to search for Poseidon Packages
482		(could be a Poseidon repository)
483	--poseidonVersion ARG	Poseidon version the packages should be updated to:
484		e.g. "2.5.3" (default: Nothing)
485	--ignorePoseidonVersion	Read packages even if their poseidonVersion is not
486		compatible with the trident version. The assumption
487		is, that the package is already structurally adjusted
488		to the trident version and only the version number is
489		lagging behind.
490	--versionComponent ARG	Part of the package version number in the
491		POSEIDON.yml file that should be updated: Major,
492		Minor or Patch (see <a href="https://semver.org">https://semver.org</a> )
493		(default: Patch)
494	--noChecksumUpdate	Should update of checksums in the POSEIDON.yml file
495		be skipped
496	--ignoreGeno	ignore SNP and GenoFile
497	--newContributors ARG	Contributors to add to the POSEIDON.yml file in the
498		form "[Firstname Lastname](Email address);..."
499	--logText ARG	Log text for this version jump in the CHANGELOG file
500		(default: "not specified")
501	--force	Normally the POSEIDON.yml files are only changed if
502		the poseidonVersion is adjusted or any of the
503		checksums change. With --force a package version
504		update can be triggered even if this is not the case.

505 It can be called with a lot of optional arguments

```
506 trident update -d ... -d ... \
507     --poseidonVersion "X.X.X" \
508     --versionComponent Major/Minor/Patch \
509     --noChecksumUpdate
510     --ignoreGeno
511     --newContributors "[Firstname Lastname](Email address);..."
```

```
512 --logText "short description of the update"
513 --force
```

514 By default `update` will not edit a package's POSEIDON.yml file, even when arguments like `--versionComponent`,  
515 `--newContributors` or `--logText` are explicitly set. This default exists to run the function on a large set of  
516 packages where only few of them were edited and need an active update. A package will only be modified by  
517 `update` if either

- 518 • any of the files with checksums (e.g. the genotype data) in it were modified,
- 519 • the `--poseidonVersion` argument differs from the `poseidonVersion` in the package's POSEIDON.yml  
520 file
- 521 • or the `--force` flag was set in `update`.

522 If any of these applies to a package in the search directory (`--baseDir/-d`), it will be updated. This includes  
523 the following steps:

- 524 • If `--poseidonVersion` is different from the `poseidonVersion` field in the package, then that will be  
525 updated.
- 526 • The `packageVersion` will be incremented. If `--versionComponent` is not set, then it falls back to `Patch`,  
527 so a change in the last position of the three digit version number. `Minor` increments the middle, and `Major`  
528 the first position (see [semantic versioning](#)).
- 529 • The `lastModified` field will be updated to the current day (based on your computer's system time).
- 530 • The contributors in `--newContributors` will be added to the `contributor` field if they're not there already.
- 531 • If any checksums changed, then they will be updated. If certain checksums are not set yet, then they will  
532 be added. The checksum update can be skipped with `--noChecksumUpdate` or partially skipped for the  
533 genotype data with `--ignoreGeno`.
- 534 • The CHANGELOG.md file will be updated with a new row for the new version and the text in `--logText`  
535 (default: "not specified"), which will be appended as the first line of the file. If no CHANGELOG.md file  
536 exists, then it will be created and referenced in the POSEIDON.yml file.

537 :heavy\_exclamation\_mark: As `update` reads and rewrites POSEIDON.yml files, it may change their inner order,  
538 layout or even content (e.g. if they have fields which are not in the [Poseidon package definition](#)). Create a backup  
539 of the POSEIDON.yml file before running `update` if you are uncertain.

## 540 1.3 Inspection commands

### 541 1.3.1 List command

542 `list` lists packages, groups and individuals of the datasets you use, or of the packages available on the server.

543 Click here for command line details

```
544 Usage: trident list ((-d|--baseDir DIR) | --remote [--remoteURL ARG])
545                 (--packages | --groups | --individuals
546                 [-j|--jannoColumn JANNO_HEADER]) [--raw]
```

547 List packages, groups or individuals from local or remote Poseidon  
548 repositories

549 Available options:

```
551 -h,--help          Show this help text
552 -d,--baseDir DIR    a base directory to search for Poseidon Packages
```

```

553             (could be a Poseidon repository)
554  --remote      list packages from a remote server instead the local
555                 file system
556  --remoteURL ARG URL of the remote Poseidon server
557                 (default: "https://c107-224.cloud.gwdg.de")
558  --packages    list all packages
559  --groups      list all groups, ignoring any group names after the
560                 first as specified in the Janno-file
561  --individuals list individuals
562  -j,--jannoColumn JANNO_HEADER
563                 list additional fields from the janno files, using
564                 the Janno column heading name, such as Country, Site,
565                 Date_C14_Uncal_BP, Endogenous, ...
566  --raw         output table as tsv without header. Useful for piping
567                 into grep or awk
568  --ignoreGeno  ignore SNP and GenoFile

```

569 To list packages from your local repositories, as seen above you can run

```
570 trident list -d ... -d ... --packages
```

571 This will yield a table like this

```

572 .------.------.------.
573 |           Title           |    Date    | Nr Individuals |
574 :=====:=====:=====:
575 | 2015_1000Genomes_1240K_haploid_pulldown | 2020-08-10 | 2535          |
576 | 2016_Mallick_SGDP1240K_diploid_pulldown | 2020-08-10 | 280           |
577 | 2018_BostonDatashare_modern_published   | 2020-08-10 | 2772          |
578 | ...                                     | ...         |               |
579 '-----'-----'-----'

```

580 so a nicely formatted table of all packages, their last update and the number of individuals in it.

581 To view packages on the remote server, instead of using directories to specify the locations of repositories on  
582 your system, you can use `--remote` to show packages on the remote server. For example

```
583 trident list --packages --remote
```

584 will result in a view of all published packages in our [public online repository](#).

585 You can also list groups, as defined in the third column of EIGENSTRAT `.ind` files (or the first column of a  
586 PLINK `.fam` file), and individuals with `--groups` and `--individuals` instead of `--packages`.

587 The `--individuals` flag provides a way to immediately access information from the `.janno` files on the  
588 command line. This works with the `-j/--jannoColumn` option. For example adding `--jannoColumn Country`  
589 `--jannoColumn Date_C14_Uncal_BP` to the commands above will add the `Country` and the `Date_C14_Uncal_BP`  
590 columns to the respective output tables.

591 Note that if you want a less fancy table, for example because you want to load this into Excel, or pipe into  
592 another command that cannot deal with the neat table layout, you can use the `--raw` option to output that  
593 table as a simple tab-delimited stream.

### 594 1.3.2 Summarise command

595 **summarise** prints some general summary statistics for a given poseidon dataset taken from the .janno files.

596 [Click here for command line details](#)

597 Usage: trident summarise (-d|--baseDir DIR) [--raw]

598 Get an overview over the content of one or multiple Poseidon packages

599 Available options:

601 -h,--help	Show this help text
602 -d,--baseDir DIR	a base directory to search for Poseidon Packages (could be a Poseidon repository)
604 --raw	output table as tsv without header. Useful for piping 605 into grep or awk

606 You can run it with

607 trident summarise -d ... -d ...

608 which will show you context information like – among others – the number of individuals in the dataset, their  
609 sex distribution, the mean age of the samples (for ancient data) or the mean coverage on the 1240K SNP array  
610 in a table. **summarise** depends on complete .janno files and will silently ignore missing information for some  
611 statistics.

612 You can use the --raw option to output the summary table in a simple, tab-delimited layout.

### 613 1.3.3 Survey command

614 **survey** tries to indicate package completeness (mostly focused on .janno files) for poseidon datasets.

615 [Click here for command line details](#)

616 Usage: trident survey (-d|--baseDir DIR) [--raw]

617 Survey the degree of context information completeness for Poseidon packages

618 Available options:

620 -h,--help	Show this help text
621 -d,--baseDir DIR	a base directory to search for Poseidon Packages 622 (could be a Poseidon repository)
623 --raw	output table as tsv without header. Useful for piping 624 into grep or awk

625 Running

626 trident survey -d ... -d ...

627 will yield a table with one row for each package. See trident survey -h for a legend which cell of this table  
628 means what.

629 Again you can use the --raw option to output the survey table in a tab-delimited format.



### 630 1.3.4 Validate command

631 `validate` checks poseidon datasets for structural correctness.

632 Click here for command line details

633 Usage: `trident validate (-d|--baseDir DIR)`

634 Check one or multiple Poseidon packages for structural correctness

635

636 Available options:

637	<code>-h,--help</code>	Show this help text
638	<code>-d,--baseDir DIR</code>	a base directory to search for Poseidon Packages
639		(could be a Poseidon repository)
640	<code>--ignoreGeno</code>	ignore SNP and GenoFile
641	<code>--noExitCode</code>	do not produce an explicit exit code
642	<code>--ignoreDuplicates</code>	do not stop on duplicated individual names in the
643		package collection

644 You can run it with

645 `trident validate -d ... -d ...`

646 and it will either report a success (`Validation passed`) or failure with specific error messages to simplify fixing  
647 the issues.

648 `validate` tries to ensure that each package in the dataset adheres to the [schema definition](#). Here is a list of  
649 what is checked:

- 650 • Presence of the necessary files
- 651 • Full structural correctness of `.bib` and `.janno` file
- 652 • Superficial correctness of genotype data files. A full check would be too computationally expensive
- 653 • Correspondence of BibTeX keys in `.bib` and `.janno`
- 654 • Correspondence of individual and group IDs in `.janno` and genotype data files

655 In fact much of this validation already runs as part of the general package reading pipeline invoked for many  
656 trident subcommands (e.g. `forge`). `validate` is meant to be more thorough, though, and will explicitly fail if  
657 even a single package is broken.

658 Remember to run it with `--logMode VerboseLog` to get more information if the output is not sufficient to debug  
659 an issue.