Genotype Pattern Mining for Human Digenic Traits

Jurg Ott - 2 Jan 2025   Maintained with *KompoZer*

This document describes a program package, *GPM* [10], for finding pairs of genotypes or of DNA variants associated with a genetic trait (digenic disease), where each genotype or variant by itself may not be disease associated. *GPM* consists of two algorithms, *Gpairs* and *Vpairs*, described below. These programs are available for Windows and Linux (Ubuntu) but they perform better in Linux than Windows. In particular, the latest feature implemented here runs only in Linux because it uses the Linux sort routine: After running a specific test like the Fisher test (see below), output is sorted by a quantity of choice, e.g., Fisher chi-square or the odds ratio.

If you want to use Pascal, in a native Ubuntu environment, simply type fpc to see whether the Free Pascal compiler is already installed. If it is not, follow instructions issued by Ubuntu on how to install it. However, Pascal is not needed to use the (pre-compiled) *Gpairs* and *Vpairs* programs described below.

Published datasets in *plink* format, previously obtained in my lab, are also available for download. When you use them in a publications please acknowledge the paper that described the dataset first.

Note: The current program version is 3.15 (15 Dec 2024). Versions older than 2.64 (24 Feb 2024) contained a bug in the calculation of the 2-sided Fisher-Irwin exact test, which is no longer present in the current program version.

Ubuntu running within Windows and the Free Pascal compiler, fpc

Ubuntu Linux can run within Windows, which represents a very easy way of running the two operating systems on the same Windows computer. Installation instructions on the Microsoft webpage will install an older fpc version. Steps outlined below are based on an Ubuntu website. Follow these steps to install the latest Ubuntu and fpc versions. Type `cmd` in your Search box and right-click on the Command Prompt app to run it as an Administrator. To see a list of available Linux distributions, type

```
wsl --list --online
```

To install version (22.04), type

```
wsl --install -d Ubuntu-22.04.
```

In my case, installation of version 22.04 started but ended with a note saying that installation had failed. However, Ubuntu turned out to be present in the list of all apps and was working fine. Next, start the Ubuntu app and type, in Ubuntu,

```
sudo apt update.
```

There will be a long list of packages. Then type

```
sudo apt upgrade
```

and press Enter so all packages can be upgraded.

At this point, you may want to install the latest fpc version but this is not essential for the rest of this page. Type `fpc` and Linux will say that fpc was not found but can be installed with a specific command. Type the proposed command, which will install the latest version, 3.2.2, of fpc. Be sure this is the latest version and not an older version! Now type

```
sudo apt install gcc.
```

After this step, you should be able to successfully use fpc although it may issue some warnings. To keep these from occurring, type

```
sudo nano /etc/fpc.cfg
```

and add the following line in the search path for libraries,

```
-Fl/usr/lib/gcc/x86_64-linux-gnu/11/
```

Frequent Pattern Mining

Frequent Pattern Mining (FPM) methods can rapidly pick frequent patterns from large databases of items. The first such approach was implemented in the *Apriori* program [3, 4], which was developed to mine consumer data. In addition to finding frequent items commonly purchased by a consumer, *Apriori* also furnishes **association rules**, that is, estimates of P(Y|X), where Y and X are purchased goods like bread, milk, and wine. Such association rules can predict how likely it is that someone buying X will also buy Y. Further information may be found in our recent reviews [5-7].

Here we apply FPM principles to genetic case-control studies with a moderate number of DNA variants and individuals. Specifically, we consider pairs of genotypes (genotype patterns) with one genotype each from two variants. Genotypes are labeled 1, 2, and 3 for AA, AB, and BB, respectively, and the label 0 (zero) stands for "missing". For example, a genotype pattern might be X = AB–AA = 2–1. Phenotypes are labeled 2 for cases and 1 for controls, and we want to find association rules, P(Y=2|X), that is, estimates for the conditional probability (based on the proportion) of being a case among all individuals with genotype pattern X. In FPM terminology [3, 4], P(Y|X) is referred to as **confidence**, and P(X) is the **support** for X. In statistics, *confidence* is known as the (positive) *predictive value*.

In a previous publication [8], we applied the *Apriori* algorithm to find interaction (epistasis) effects of variants in case-control data. Here, however, the focus is simply on frequencies of genotype patterns and whether they are different in cases and controls [10] while individual variants may not show such differences. Other approaches to find epistatic variants exist [5].

Pairs of variants and genotypes

Consider two variants (SNPs), possibly on different chromosomes. Each variant has 3 genotypes, so there are 3 3 = 9 genotype pairs. All possible variant pairs are numbered 1, 2, ..., M, where M may be a rather large number. For a given pair of variants, we proceed in two ways as follows, where the two programs discussed below assume that the data are in standard *plink* format [9].

Vpairs program. For each of cases and controls, a 3 x 3 table of genotype pairs is set up as shown in the example below.

```
CONTROLS
       |   SNP 2
SNP 1  |  1  2  3
-------+---------
 1     | 14  9 14
 2     | 19 55 29
 3     | 13 31 16
-----------------

CASES
       |   SNP 2
SNP 1  |  1  2  3
-------+---------
 1     |  8 31  6
 2     | 17 50 31
 3     | 10 30 17
-----------------
```

Then a likelihood ratio test is carried out to test for differences in interaction between cases and controls, which will result in a chi-square value with 4 df as shown below for the example data. The expression for the latter is given by C(het) = C(cases) + C(controls) - C(cases and controls combined), where C refers to likelihood ratio chi-square (if computed as Pearson chi-square, results will be approximate). See the *conting* program in my Statistical Genetics Utilities.

```
Heterogeneity analysis, genotype tests
          chi-sq df   p
CONTROLS 11.0769  4 0.0257
CASES     6.5365  4 0.1625
BOTH      2.9578  4 0.5649
Heterog  14.6556  4 0.0055
```

Each of the *M* variant pairs furnishes such a chi-square result. The *Vpairs* program will read on the command line a number of input parameters (just type `Vpairs` to see what parameters need to be furnished on the command line), one of which is the number of threads the program should use. It will then assign an equal number of variant pairs to each thread for analysis.

Gpairs program. For two given variants, this program will work on each of the 9 genotype pairs. For each genotype pair (pattern), X, the number of cases and controls with and without the pattern will be determined, which leads to a table as shown below, where *a*, *b*, *c*, and *d* refer to numbers of individuals:

Phenotype	X present	X absent
cases	*a*	*b*
controls	*c*	*d*
	N_X	N_{noX}

The numbers *a*, *b*, *c*, and *d* form a 2 x 2 table, for which one of the following statistics may be computed:

- Pearson chi-square (C), 2-sided test. Tables with expected numbers smaller than 2 in any cell will be skipped..
- Fisher's exact test (F), one-sided mid-p-value (association of cases with presence of X).
- Odds ratio (O) in table above
- Prediction accuracy (A)

Each pattern, X, will need to pass some filters, that is, a pattern may be skipped when (depending on user input):

- its two variants are on the same chromosome.
- an expected number in the 2 x 2 table falls below a threshold like 1 (only applies to chi-square test).
- the support, $a + c$, is smaller than a threshold like 20.
- the confidence, $C = a/(a + c)$, is smaller than a threshold like 0.95. No check on confidence if threshold = 0. It is usually best not to condition on confidence.

Note the following conventional expressions:

- a = true positives
- d = true negatives
- b = false negatives
- c = false positives
- $a/(a + b)$ = sensitivity or power
- $d/(c + d)$ = specificity
- $a/(a + c)$ = positive predictive value, confidence
- $d/(b + d)$ = negative predictive value
- $ad/(bc)$ = odds ratio
- $(a + d)/(a + b + c + d)$ = prediction accuracy

Type `Gpairs` to see what parameters to enter on the command line. Specifying a high minimum confidence of, say, 95% will select patterns with $a/(a + c) \geq 0.95$. To avoid conditioning on confidence (recommended), set the minimum confidence equal to zero.

Statistical testing. The chi-square value, either (1) obtained from the 2 x 2 table above or (2) transformed from the Fisher *p*-value, is large when a genotype pattern frequency differs much between cases and controls. The associated *p*-value represents the empirical significance level on a per variant basis. For a significance level corrected for testing multiple variants, due to the large computational burden, there is currently no provision for permutation testing. Instead, Bonferroni correction is applied to obtain relevant corrected *p*-values. One might consider pruning the data to a set of relatively independent variants, for example, with the *plink* command, `--indep 50 5 2`, but this may eliminate important variants.

The chi-square test (C) is 2-sided by design, that is, chi-square will be large when the frequency of X is higher in cases than controls, or higher in controls than cases. However, 2-sided testing is kind of an "over-kill"; any single one of the 9 genotype pairs will be tested so that, if one genotype pair is much more frequent in cases than controls, there must be genotype patterns that are more frequent in controls than cases. Thus, the recommended test is the classical **Fisher (1-sided) exact test** (F). Alternatively, if the focus is on prediction, the odds ratio statistic seems appropriate.

Gpairs program usage. Download the Gpairs programs file into a suitable folder (directory), for example, C:\prog (Windows) or ~/bin (Linux). The program files are *Gpairs* (Linux) and *Gpairs.exe* (Windows), currently allowing for up to 900,000 variants and 5,000 individuals. Smaller program versions are *Gpairs5* and *Gpairs5.exe*, which can handle up to 500,000 variants and 3,000 individuals.
The programs are preferably run in a command window (terminal). In Windows, you may type `cmd` in the search box, which will open a command window. Then, change to the program folder by typing, for example, `cd C:\prog`.

In Linux, it may be necessary to make the programs executable by typing, for example, `chmod +x Gpairs5`. Your data files should be in standard *plink* format [9], for example, *AMD.map* and *AMD.ped*. To carry out a sample analysis, you will need to type the program name followed by some command line parameters. The required list of command line parameters will be displayed when you just type the program name, for example, *Gpairs5*, which will result in the following list:

```
Gpairs program version 3.15 (15 Dec 2024)  https://github.com/jurgott/gpm_prog
(C) 2021-2024 Jurg Ott   GNU General Public License v3

Maximum program constants are as follows:
     3,000 individuals
   500,000 variants, SNPs
        70 threads, CPUs
Use with the following commandline parameters:
 1 prefix
 2 plink format dataset name without .map and .ped
 3 variants in a pair must be on different chromosomes if 1
 4 number of threads to use, >2
 5 minimum test statistic for output
 6 Minimum support (eg. 20) for testing
 7 Minimum confidence (eg. 0.80) for testing; set =0 to disregard (recommended)
 8 Test stat: 1 = Chisquare; 2 = Fisher; 3 = odds ratio; 4 = accuracy
 9 folder where report file is written, eg. /home/joe/

For example, to run in the background, Linux:
 (Gpairs  T- AMD 0 5 20 20 0 2 /home/joe/ >/dev/null)&
```

You will need to know how many threads your computer has. In Linux, you see this after typing `nproc`.

Parameter #5 indicates how many genotype pairs will be shown in the output file. That number depends on the test statistic. For example, if test stat = odds ratio and parameter #5 = 5.5, only those genotype pairs with OR > 5.5 will be output.

Parameter #6 (*minsupp*) says that genotype pairs occurring in fewer than *minsupp* individuals will be skipped. Larger values of *minsupp* will reduce the testing burden, i.e. make Bonferroni correction less stringent.

Parameter #7 (*minconf* as a proportion of individuals) puts a limit on confidence (*conf*), which is the proportion of cases among individuals with a given genotype pair. This parameter says that genotype pairs occurring in fewer than a proportion of *minconf* individuals will be skipped. Large values of *minconf* like 0.90 may not reduce the testing burden and only lead to an enrichment of genotype pairs with high confidence. Practice shows that *minconf* = 0 (no restriction on *conf*) is usually the best choice, or choose a value of *minconf* equal to the proportion of cases among all individuals, which is the confidence achieved at random. Earlier program versions used *minconf* as a percentage like 90.

Running the AMD dataset in Linux with minimum support of 20 (at least 20 individuals will need to carry a given genotype pattern or else it will be skipped), using 10 threads, and no restriction on confidence will be accomplished by typing

```
Gpairs5 Sample AMD 1 10 20 20 0 1 5 /home/joe/
```

assuming the login is joe. This command will run a (2-sided) chi-square test, that is, genotype patterns more frequent in cases or in controls will be of interest. To perform this run in the background without output to the screen, type

```
(Gpairs5 Sample AMD 1 10 20 20 0 1 5 /home/joe/ > /dev/null)&
```

However, it is a good idea to initially run a command in the foreground so you see potential error messages before submitting the job in the background. Once a (long) job has started in the foreground and is invoking multiple threads (in Linux, the `top` command will show you that %CPU exceeds 100), it usually runs fine and you may interrupt it with Ctrl-C and submit it in the background. In Linux, for a program running in the background, you stop it with the `kill` command.

While the *Gpairs* program is running, it will produce, for each of the threads, two temporary files with names starting with T. **Please don't touch these files!** They will be automatically deleted after successful program termination. You may monitor program execution with the *top* command in Linux. Also, in the folder specified on command line parameter #10, you will see a report file with a name ending in `rpt`, which you may inspect by typing `cat *rpt`. This file reports progress of thread #1, which may end early or late compared with other threads. The projected time for thread #1 to finish is only a very approximate estimate for the program to finish. To verify that *Gpairs* has finished, type `top`, where you should no longer see *Gpairs* listed as one of the running procedures.

If the *Gpairs* program terminates and files starting with T (as many as the number threads requested) still exist, something has gone wrong. Most likely, the program has run out of memory. In this case, one potential remedy is to restart the program but you request fewer threads as each thread requires an amount of memory like 1GB or more. *Gpairs* will then run slower but require a smaller total amount of memory. It is good practice to check memory usage from time to time with the *top* command.

In Linux, the *Gpairs* program will produce three output files (only the first two files in Windows) as follows: (1) A log file providing information on the run; (2) a file ending in "out" that will contain all genotype pairs requested (possibly > 1 mio); and (3) a file ending in "sorted", which will contain up to 100,000 of the genotype pairs with largest test statistic.The "sorted" output file may be directly read into *Excel* or some other spreadsheet program like *Libreoffice/calc* (the latter is available in Ubuntu). This file is usually all you need, so you may safely discard the "out" file, which can be very large when it contains many more than 100,000 lines.

Column headers in the output files of the *Gpairs* program

- chisq = chi-square value, transformed to have 2 df, obtained by Fisher exact test or chi-square test
- Y = 2 if OR > 1, Y = 1 if OR $\leq$ 1
- pEmp = empirical significance level associated with chi-square, uncorrected for multiple testing
- pBon = Bonferroni-corrected significance level based on the number of tests performed, see output log file
- supp = support, number of individuals with the given pattern, equal to $a + c$ (see above)
- conf = confidence, proportion of cases among those with the given pattern; = a/supp if Y=2, = c/supp if Y=1.
- a, b, c, d = number of individuals in 2 x 2 table discussed above
- OR = odds ratio; use $(a + 0.5)(d + 0.5)/[(b + 0.5)(c + 0.5)]$ if $ad/(bc)$ fails. If test stat = 4, ACC will be reported instead of OR.
- v1 = sequential number of first SNP in genotype pair = line number of first SNP in map file
- ch1 = chromosome number of first SNP
- rs1 = ID of first SNP; no SNP should have a dot as its ID (see `nodots` program)
- bp1 = basepair position of first SNP
- g1 = genotype (1, 2, 3) at first SNP
- v2, ch2, rs2, bp2, g2 => analogous for second SNP

Utility programs

Two utility programs are included in this package.

In some datasets, variants without a known name (rs ID) are sometimes identified by a dot so that all such variants carry the same identifier. In *plink* filesets, the `nodot` program replaces each such dot with a unique name.

If a particular pair of genotypes looks interesting, the `pairSNPs` program will provide detailed information about the two variants involved.

References

1. R. J. Klein, C. Zeiss, E. Y. Chew, J. Y. Tsai, R. S. Sackler, C. Haynes A. K. Henning, J. P. SanGiovanni, S. M. Mane, S. T. Mayne, M. B. Bracken, F. L. Ferris, J. Ott, C. Barnstable and J. Hoh: Complement factor H polymorphism in age-related macular degeneration. *Science*, 308(5720), 385-9 (2005) doi: 10.1126/science.1109557
2. A. Dewan, M. Liu, S. Hartman, S. S. Zhang, D. T. Liu, C. Zhao, P. O. Tam, W. M. Chan, D. S. Lam, M. Snyder, C. Barnstable, C. P. Pang and J. Hoh: HTRA1 promoter polymorphism in wet age-related macular degeneration. *Science*, 314(5801), 989-92 (2006) doi: 10.1126/science.1133807
3. R. Agrawal, T. Imielinski and A. Swami: Mining association rules between sets of items in large databases. In: *ACM SIGMOD Conference on Management of Data*. Washington DC (1993) doi: 10.1145/170035.170072
4. R. Agrawal and R. Srikant: Fast algorithms for mining association rules. In: *20th VLCB Conference*. Proceedings of the 20th VLCB Conference, Santiago, Chile (1994) URL: https://www.vldb.org/conf/1994/P487.PDF
5. A. Okazaki, S. Horpaopan, Q. Zhang, M. Randesi and J. Ott: Genotype pattern mining for pairs of interacting variants underlying digenic traits. *Genes*, 12(8), 1160 (2021) doi: 10.3390/genes12081160
6. A. Okazaki and J. Ott: Machine learning approaches to explore digenic inheritance. *Trends Genet* (2022) doi: 10.1016/j.tig.2022.04.009
7. J. Ott and T. Park: Overview of frequent pattern mining. *Genomics Inform*, 20(4), e39 (2022) doi: 10.5808/gi.22074
8. Q. Zhang, Q. Long and J. Ott: AprioriGWAS, a new pattern mining strategy for detecting genetic variants associated with disease through interaction effects. *PLoS Comput Biol*, 10(6), e1003627 (2014) doi: 10.1371/journal.pcbi.1003627
9. C. C. Chang, C. C. Chow, L. C. Tellier, S. Vattikuti, S. M. Purcell and J. J. Lee: Second-generation PLINK: rising to the challenge of larger and richer datasets. *Gigascience*, 4, 7 (2015) doi: 10.1186/s13742-015-0047-8
10. Q. Zhang et al: A multi-threaded approach to genotype pattern mining for detecting digenic disease genes. *Front Genet* 14, 1222517 (2023) doi: 10.3389/fgene.2023.122251