srashkin

A (not so) Short Intro to Plink v2 and Running GWAS with it

A few notes to begin

Since the QC-ed version of the ukbiobank data will be stored in plink v2 binary format (not the same as plink v1.9 binary format), most analyses will probably be done in plink v2. While plink v2 is not yet in a stable release version, it has some features that are really nice. However, there are still some issues it has in actually running, which require some special handling.

Nice parts of plink v2

No need for special handling of dosage data! It will automatically read in dosage data and use it if it is there. There are a few commands that I think may still be incompatible with dosage data, in which case that part of the data will be stripped.

You can also do all the normal data filtering on dosage data that was not possible in plink v1.9. In other words, arguments like `--maf`, `--mind`, `--geno`, and `--hwe` all work. Slight caveat with `--hwe`, though, is that, unlike plink v1.9, it does not automatically filter to just controls in case/control data.

You can now specify multiple phenotypes to be analyzed for dosage data. You just submit either a `psam` file (the new plink v2 binary file equivalent of a `fam` file) with multiple columns for each phenotype or submit a phenotype file with the `--pheno` command and specify which columns using the `--pheno-name` command. As a note, each phenotype will be analyzed in sequence rather than analyzing them all simultaneously.

More commands in plink v2 allow for multithreading, which should speed up most processes. In fact, as opposed to plink v1.9, many if not most commands default to a multithreaded implementation. Which leads to the...

Bad parts of plink v2

Plink v2 does not always seem to communicate with the scheduler in terms of memory and processor resources. If you do not specifically tell plink how much memory and how many threads it is allowed, it will bypass whatever your submission script has requested and try to use the entire node's resources. Which is **VERY BAD**. When we were first starting to run things using plink v2, we managed to crash our nodes a few times before realizing that these plink v2 jobs where trying to use all 64 CPUs on a node even if we had only allocated 1. Hopefully, this will be fixed in future versions, but, for now, you absolutely have to use `--memory [MB]` and `--threads [# threads]` to prevent plink v2 from exceeding what has been allocated. If you don't use `--memory`, you will most likely just get messages that your job has exceeded the amount of memory requested from the scheduler and it will never run since it is always going to try to access more than it should. If you do not use `--threads`, your job will run, but it will do so on all 64 CPUs and possibly end badly for our nodes.

A further note is that sometimes you have to request more memory from the scheduler than with `--memory` to prevent plink v2 from going over what has been requested. For instance, I've had to request `#PBS -l vmem=6GB` paired with `--memory 5000`.

As a last note: supposedly, character values should be allowed as covariates (ie, "yes" and "no") and dummy coding will occur internally. But, when I tried this on the current version, I was getting errors. Maybe when the stable version is released this won't be the case.

Running GWAS with Plink v2

With those caveats out of the way, we can run a GWAS using plink v2. There is a version in the lab's software directory: `/wittelab/data1/software/bin/plink2`.

There is online documentation, but that is also not completely finished and does not describe all the available options. You can view the full list of options currently available via the help menu: `/wittelab/data1/software/bin/plink2 --help`. But there is still limited functionality beyond running GWAS and, to do some things, you may have to down-convert to plink v1.9 and run it there.

Convert to new binary format

This step is not, strictly speaking, 100% necessary. However, if you load data in a different format, plink v2 will automatically convert it prior to doing anything else. It just does not save it unless instructed. If you are working with the ukbiobank data, this step has already been done and you can skip this step.

If you do not already have pgen/psam/pvar files, plink v2 can read in any of the following file types as well:

- Plink 1 binary (bed/bim/fam)
  - Use `--bfile [prefix]` or `--bed [filename]`, `--bim [filename]`, `--fam [filename]`
- VCF
  - Use `--vcf [filename] <dosage=[field]>` or `--bcf [filename]`
- Oxford-format genotype
  - Use `--data [prefix]` or `--bgen [filename]`, `--sample [filename]`
- Plink 1 dosage (anything you could read in with the plink v1.9 `--dosage` command)
  - Use `--import-dosage [filename]`

There are more options for several of those types, but those are all listed in the online documentation already, so I am not going to go through them. (If your data is in any other format, you'll have to use plink v1.9 to convert to one of these first before using plink v2. This should change in the future.)

You can then convert to the new binary format by using the command `--make-pgen`.

Here is an example using the ukbiobank raw bgen files:

```
path="/wittelab/data1/srashkin/ukbiobank/bgen"

/wittelab/data1/software/bin/plink2 \
--bgen $path/ukb_imp_chr1_v2.bgen \
--sample $path/ukb1410_imp_chr1_v2_s487406.sample \
--memory 2000 \
--threads 1 \
--make-pgen \
    --out $path/ukb.chr1
```

Prepare phenotype and covariate files

Prepare as you would with plink v1.9. As opposed to plink v1.9, plink v2 will default to analyzing all phenotypes in the psam and pheno files. So your phenotype file has to be space- or tab-delimited and starts with FID and IID, but can be followed by any number of phenotypes, where different individuals can have non-missing values for each phenotype. You can specify which columns you want to analyze using `--pheno-name` or `--pheno-col-nums`, whether you specify a separate phenotype file or if you supply a psam file with extra columns for the different phenotypes. Your phenotypes can be binary or continuous or a mix. One command will be used to run all the GWAS, and plink v2 will automatically determine whether it is binary or continuous. Missing values can be encoded as -9 as in plink v1.9 or as NA/nan. Like plink v1.9, case/control data is expected to be coded as 1=control, 2=case, but you can still use `--1` to tell plink your phenotypes are coded as 0=control, 1=case.

Your covariate file should be prepared in the same way: space- or tab-delimited starting with FID and IID followed by any covariates you want. Supposedly, you can include categorical covariates, but I still got errors when I tried, so found it safer to just manually recode to indicator or dummy coding.

Run GWAS

As opposed to plink v1.9 where logistic and linear regression were run with different commands, since plink v2 allows for multiple phenotypes, there is now just one command for both: `--glm`.

Here is an example using the QC'd ukbiobank data:

```
    genpath="/wittelab/data1/srashkin/ukbiobank/GWAS/qcdata/chr1"
    ppath="/wittelab/data1/srashkin/ukbiobank/GWAS/phenofiles"

    /wittelab/data1/ssoftware/bin/plink2 \
    --pfile $genpath/ukb.qc.v5.chr1.genomaf.hwe.12122017 \
    --pheno $ppath/allcancer.12122017.phen \
    --pheno-name bladder-colorectal \
    --1 \
    --covar $ppath/allcancer.12122017.cov \
    --covar-name PC1-PC10,age_assessment,sex_self,genotyping.array \
    --glm \
    --memory 5000 \
    --threads 5 \
    --out $opath/ukb.chr1.12122017
```

If you would like to add an additional MAF filter, you can do it in this step by adding `--maf [minimum freq]`, just like in plink v1.9, but it now also works for dosage data with no special handling. If you would like to restrict to a specific subset of SNPs (ie, if you want to add an additional filter for the imputation INFO score), you can use `--extract` or `--exclude` as in plink v1.9 to include or exclude a list of SNPs.

If you are using ukbiobank data, the QC'd data will be filtered on INFO score > 0.3, but all SNPs will be included, regardless of MAF, so you will probably want to at least use --maf.

You will get two files per phenotype. If you specified `--out myoutput` and are analyzing pheno1, a case/control variable, you will get:

1. myoutput.pheno1.glm.logistic.id - the list of samples used in the analysis of that phenotype.
2. myoutput.pheno1.glm.logistic - the results of the GWAS.

Presumably if one of your variables is continuous, the output file names will have "linear" instead of "logistic", but I have not tested that.