RICOPILI HOW TO post-imputation module on summary statistics

(including aligning external sumstats)

Gio Panagiotaropoulou , Ph.D. student, gpanagio@broadinstitute.org
Prof. Stephan Ripke, M.D., Ph.D., Group Leader, stephan.ripke@charite.de
GResU (GWAS Research Unit, BIH, Berlin)
(Last modified on: 16.09.2019)

There are a couple of reasons why you might want to do that:

- It's a pretty efficient way to create multiple leave-one-out meta-analyses.
- You can integrate external data, whose raw genotypes you don't have access to.
- You want to get the full set of publication - ready output files even from just one sumstat file. (manhattan/qq/region/forest-plots, excel output files)

We highly recommend reading the general "daner" description in parallel, since there is a lot of complementary information.

Please make sure, that all columns in the "daner" file (see above) are present, otherwise there are some Ricopili Modules, which faile. If you don't have this information, feel free to fill with dummy variables (e.g. "ngt" or "INFO").

Another point to consider are removing duplicated SNP names in your sumstats. That can frequently happen when you get these files from outside sources.

How to run the post-imputation module on summary statistics result files (using the --results option).

- Copy (or make symbolic links of) the sumstat files of single cohorts and/or meta-analyses ("`daner*gz`" files) into an empty working directory.
    - Format needs to be in "`daner`" format, see here.
- Create a file with the list of summary statistic files to be meta-analyzed by echoing all the "daner*gz" filenames of studies of interest into a text file.
    e.g. For an example dataset daner_dat1.gz, daner_dat2,gz, daner_dat3.gz, use this command:

```
ls daner_dat*.gz > dataset_dir
```

Like this you would create a textfile "`dataset_dir`" with the following contents (command:):

```
daner_dat1.gz
daner_dat2.gz
daner_dat3.gz
```

- Copy the "`reference_info`" file from a previous analysis into your working directory.[1] In the first row it will point to an imputation reference, to be used for LD information (e.g. for clumping procedures). Optimally this is the imputation reference that was used when creating the dosages/genotypes used for the sumstats. If this is not the case please see below for tips to align Ricopili-external sumstats.
- Run the "`postimp_navi`" command with the "`--result dataset_dir`" option, also specifying an output name ("`--out`") and any other appropriate flag you wish.
- If finished, the same output files as for meta-analyses of dosage data will be found in the corresponding output directory under "`distribution/`".
- The pipeline steps are listed here. In case you want to repeat these steps several times (e.g. for leave-one-out analyses), you should pay attention to the following:
  - First the sumstat files are split into genomic chunks (step "`chunk`"). This is done only once for each input (daner) file.
  - Please wait for the above step to be finished once for all input files, then
  - Subsequent postimp runs can run in parallel in the same directory with distinct --out flags.

single-cohort vs. meta-analysis sumstat input files:

You can use sumstat files coming out of meta analyses as input files followed by a next meta analysis. This is not only sometimes necessary (since you might not have access to single cohort results) but also comes handy if you are interested in the **heterogeneity of effect between meta-analyses** (e.g. male meta-analysis vs. female meta-analysis)

- **Single cohort sumstat files** usually contain only 12 columns, the Number of cases and controls is coded into the header and all SNPs are assumed to have been calculated with this sample size. During meta-analysis Ricopili will internally calculate the (**half**) effective sample size, again valid for all SNP:
  - 4*Nca*Nco/(2*(Nca+Nco))
    - E.g. 439 cases and 1137 controls leads to an Neff_half: 633.43

  Meta Analysis: During meta analysis, the above single cohort effective sample size (or it's half) will be summed over all cohorts (where this SNP is present), as of note the true

[1] If you don't know where to pick one, we have some for LISA/BROAD/GCLOUD. Please contact us.

effective sample size is almost always smaller than the effective calculated from the sum of cases and controls, even for SNPs that are present in all cohorts.

- **Meta analytic sumstat files** contain additional columns with "`Nca`", "`Nco`", "`Neff_half`"[2] per each SNP (based on the two bullet points above it is hopefully clear now, that these differ between SNPs). If used as input, Ricopili recognizes column-header containing these terms and assigns the corresponding column to the specific SNPs. If you want to provide that from an **external source**[3], then please make sure to calculate `Neff_half` correctly (and not `Neff`), see formula above:
- As of note:
    - For a single cohort we recommend using single cohort danerfiles (with Nca and Nco coded into the header). Ricopili will take over the calculation of the effective sample size.
    - Nca and Nco are not only used for the calculation of the effective sample size but also as weights for the calculation of the resulting allele frequencies.
    - Nco is used as a weight for the calculation of the resulting INFO score.
    - ngt is summed up over all input files, this is only used for display and doesn't influence any other variables.

Additional considerations Sumstat Files from external sources:

- As you might be aware, Ricopili will use some reference data (1000Genomes, HRC) for LD information (used mainly for clumping).
- Ricopili has a very **efficient alignment process during the imputation** process so you don't need to take additional measures as long as you use the same reference for the imputation and postimp module.
- For **sumstats there is no alignment process** in place as of now. So even if you used the same imputation reference by name (e.g. 1000Genomes phase3, HRC) it is likely that some snp-names and or allele-names, especially for the special variants like Indels and multiallelic variants do differ between your sumstats and the Ricopili postimp reference (as defined in the file `reference_info`).
- It is possible to run the process without aligning, but the clumping procedure will have some confusing output and you will likely see many more index SNPs than there are in reality.
- So we highly recommend **aligning external sumstats to ricopili reference format**:
    - In the first row of `reference_info` you will be pointed to a directory, in which you find will find these files (one for each chromosome):

[2] Before version june18, Ricopili recognized "Neff" alone which led to problems when external sources calculated Neff and not Neff_half. We hope to have made that clearer now.
[3] if it's coming from Ricopili these columns will be correct

        - `*.EUR.frq2.gz`[4]
    - Based on these files, try to “translate” all SNPs in your input file to the Ricopili SNP name based on a hash of “CHR_POS_A1_A2” and “CHR_POS_A2_A1”.[5] Please also see below for a quick-and-dirty solution with a one-liner awk command.
    - Keep SNPs that you do not find a corresponding position[6] and count them.
        - If there are too many (e.g. more than 20%), try to find a reason for that (different imputation reference, different genome-build, specialized chip like the ONCO array, sequencing data) and decide if you there is a possibility of further curation.
- To start a post imputation module with --result option, the pipeline requires the location of the reference genome and looks for the file “`reference_info`” . Please create one by this command `echo refernce_dir reference_info`

****
The following commands will load some reference information into a hash. For whole genome translations this might take a lot of memory. Since awk is not perfect with reporting a lack of memory, please keep this in mind when performing these (e.g. starting an interactive job on a cluster farm or doing it by-chromosome)
If you encounter (and solve) problems, please let us know so that we can include a short FAQ below.
****

Quick-and-Dirty SNP Translation based on position:

Following is an awk one-liner to overwrite sumstat SNP-names with reference SNP names based on position, e.g. some SNPs have this naming convention (CHR_POS) and **not** rs-names. Please replace all capitalized words with the legend:

```
awk 'NR==FNR{chrpos_in = $CHRREF"_"$POSREF; s[chrpos_in]=$SNPREF;
next}{chrpos=$CHRSUM"_"$POSSUM; if (chrpos in s) {$SNPSUM=s[chrpos]} print }'
REFERENCE SUMSTAT > SUMSTAT.trans

CHRREF: column-number containing chromosome info in reference file
POSREF: column-number containing position info in reference file
SNPREF: column-number containing snpname in reference file
CHRSUM: column-number containing chromosome info in sumstat file
POSSUM: column-number containing position info in sumstat file
```

[4] There is the same file for other ancestries. But that’s not important for this purpose.
[5] You can possibly add a check for frequency as well, especially for ambiguous SNPs (AT, CG) though this is mostly not necessary for post-imputation SNPs.
[6] These SNPs will show up as “independent” since no LD information will be available.

```
SNPSUM: column-number containing snpname in sumstat file
SUMSTAT: file containing the sumstats with the non-matching
SNP-names:
REFERENCE: file containing reference SNP-name
SUMSTAT.trans: file, where SNP names have been overwritten
```

This minor change removes SNPs that are NOT found:

```
awk 'NR==FNR{chrpos_in = $CHRREF"_"$POSREF; s[chrpos_in]=$SNPREF;
next}{chrpos=$CHRSUM"_"$POSSUM; if (chrpos in s) {$SNPSUM=s[chrpos]; print}  }'
REFERENCE SUMSTAT > SUMSTAT.trans
```

Please note, this command can take a while, since it reads the whole reference information into RAM first.

Also please have here two files for download, one with 1KG reference, one with HRC reference, both in ricopili format:

Quick-and-Dirty SNP Translation based on position and alleles:[7,8]

If you are certain that genome build is aligned (e.g. positions lign up) then you can translate SNP names to the reference including allele information into account. Order of A1 and A2 does not matter here. The first command keeps the positions that are not found in the reference, the second one removes these variants (this is preferred).

```
awk 'NR==FNR{chrpos_in = $CHRREF"_"$POSREF"_"$A1REF"_"$A2REF; s[chrpos_in]=$SNPREF; next}
{chrpos1=$CHRSUM"_"$POSSUM"_"$A1SUM"_"$A2SUM; chrpos2=$CHRSUM"_"$POSSUM"_"$A2SUM"_"$A1SUM; if
(chrpos1 in s) {$SNPSUM=s[chrpos1]}; if (chrpos2 in s) {$SNPSUM=s[chrpos2]}; print }'
REFERENCE SUMSTAT > SUMSTAT.trans
```

This needs these additional columns:

```
A1REF: column-number containing allele1 in reference file
```

[7] You might want to replace indels-alleles first e.g. a SNP with alleles T/TCCC into alleles D/I

[8] Also you want to check first if the sumstats are on the same human build (e.g. the same positions) this will get clear quickly when looking at the number of successful translations

```
A2REF: column-number containing allele2 in reference file
A1SUM: column-number containing allele1 in sumstat file
A2SUM: column-number containing allele2 in sumstat file
```

This minor change removes SNPs that are NOT found:

```
awk 'NR==FNR{chrpos_in = $CHRREF"_"$POSREF"_"$A1REF"_"$A2REF; s[chrpos_in]=$SNPREF; next}
{chrpos1=$CHRSUM"_"$POSSUM"_"$A1SUM"_"$A2SUM; chrpos2=$CHRSUM"_"$POSSUM"_"$A2SUM"_"$A1SUM; if
(chrpos1 in s) {$SNPSUM=s[chrpos1] print }; if (chrpos2 in s) {$SNPSUM=s[chrpos2] print} }'
REFERENCE SUMSTAT > SUMSTAT.trans
```

Quick-and-Dirty FRQ Translation based on SNP name:

Following is an awk one-liner to overwrite sumstat frequencies with reference frequencies based on SNPname. Only SNPs found in reference are kept. Please replace all capitalized words with the legend[9]:

```
awk 'NR==FNR{s[$SNPREF]=$FRQREF; next}{snp=$SNPSUM; if (snp in s) {$FRQSUM=s[snp]} print }'
REFERENCE SUMSTAT > SUMSTAT.trans
```

```
SNPREF: column-number containing snpname in reference file
FRQREF: column-number containing frequency in reference file
SNPSUM: column-number containing snpname in sumstat file
FRQSUM: column-number containing frequency in sumstat file
SUMSTAT: file containing the sumstats with the non-matching
SNP-names:
REFERENCE: file containing reference SNP-name
SUMSTAT.trans: file, where SNP names have been overwritten
```

Quick-and-Dirty Position-Translation based on SNP name:

Following is an awk one-liner to overwrite positions (e.g. if they come from a different genomic built) with the positions in the reference. Naturally for that you need matching SNPnames (e.g. rs-names) in your sumstats. Since only SNPs found in reference are kept, it's worth doublechecking the resulting number of SNPs (should be more than 5M genome wide). Please replace all capitalized words with the legend.

[9] Be aware that this doesn't check for the allele the frequency belongs to. E.g. I would recommend using this only for MAF checks

```
awk 'NR==FNR{chrpos_in = $CHRREF" "$POSREF; s[$SNPREF]=chrpos_in; next}{if ($SNPSUM in s) {
print $0,s[$SNPSUM]}}' REFERENCE SUMSTAT > SUMSTAT.trans
```

```
CHRREF: column-number containing chromosome info in reference file
POSREF: column-number containing position info in reference file
SNPREF: column-number containing snpname in reference file
SNPSUM: column-number containing snpname in sumstat file
SUMSTAT: file containing the sumstats with the non-matching
SNP-names:
REFERENCE: file containing reference SNP-name
SUMSTAT.trans: file, where SNP names have been overwritten
```

Quick-and-Dirty Position-Translation based on SNP name and alleles:[10]

Following is an awk one-liner to overwrite positions (e.g. if they come from a different genomic built) with the positions in the reference, ***this time also checking if the alleles are correct***. Naturally for that you need matching SNPnames (e.g. rs-names) in your sumstats. Since only SNPs found in reference are kept, it's worth doublechecking the resulting number of SNPs (should be more than 5M genome wide). Please replace all capitalized words with the legend.

```
awk 'NR==FNR{chrpos_in = $CHRREF" "$POSREF; snp_in = $SNPREF"_"$A1REF"_"$A2REF;
s[snp_in]=chrpos_in; next}{snpsum1 = $SNPSUM"_"$A1SUM"_"$A2SUM; snpsum2 =
$SNPSUM"_"$A2SUM"_"$A1SUM; if (snpsum1 in s) { print $0,s[snpsum1]}; if (snpsum2 in s) { print
$0,s[snpsum2]};}' REFERENCE SUMSTAT > SUMSTAT.trans
```

```
CHRREF: column-number containing chromosome info in reference file
POSREF: column-number containing position info in reference file
SNPREF: column-number containing snpname in reference file
SNPSUM: column-number containing snpname in sumstat file
SUMSTAT: file containing the sumstats with the non-matching
SNP-names:
REFERENCE: file containing reference SNP-name
SUMSTAT.trans: file, where SNP names have been overwritten
```

This needs these additional columns:

```
A1REF: column-number containing allele1 in reference file
A2REF: column-number containing allele2 in reference file
A1SUM: column-number containing allele1 in sumstat file
A2SUM: column-number containing allele2 in sumstat file
```

[10] You might want to replace indels-alleles first e.g. a SNP with alleles T/TCCC into alleles D/I

With the command above the new position will be added at the end of the row, if you want the original columns replaced try this:

```
awk 'NR==FNR{chr_in = $CHRREF; pos_in = $POSREF; snp_in = $SNPREF"_"$A1REF"_"$A2REF;
s[snp_in]=chr_in; t[snp_in]=pos_in;; next}{snpsum1 = $SNPSUM"_"$A1SUM"_"$A2SUM; snpsum2 =
$SNPSUM"_"$A2SUM"_"$A1SUM; if (snpsum1 in s) { $CHRSUM = s[snpsum1]; $POSSUM = t[snpsum1];
print $0}; if (snpsum2 in s) { $CHRSUM = s[snpsum2]; $POSSUM = t[snpsum2]; print $0}'
REFERENCE SUMSTAT > SUMSTAT.trans
```

This needs these additional columns:

```
CHRSUM: column-number containing chromosome info in reference file
POSSUM: column-number containing position info in reference file
```

Quick-and-Dirty Position-Translation based on SNP name, introducing second alleles into the second file (e.g. necessary for correlation analysis in LDSC):[11]

Following is an awk one-liner to transfer the second allele from one file into another, e.g. if the second file comes with only the effect allele. Here the match comes from SNP name so make sure these are aligned first. Please replace all capitalized words with the legend. **(Be careful with three-allelic SNPs.)**

```
awk 'NR==FNR{snp_in_a1 = $SNPREF"_"$A1REF; snp_in_a2 = $SNPREF"_"$A2REF ;
s[snp_in_a1]=$A2REF; s[snp_in_a2]=$A1REF; next}{snpsum_a = $SNPSUM"_"$ASUM"; if (snpsum_a in
s) { print $0,s[snpsum_a]}; }' REFERENCE SUMSTAT > SUMSTAT.trans
```

```
SNPREF: column-number containing snpname in reference file
A1REF: column-number containing allele1 in reference file
A2REF: column-number containing allele2 in reference file
ASUM: column-number containing allele in sumstat file
SNPSUM: column-number containing snpname in sumstat file
SUMSTAT: file containing the sumstats with the single allele SNPs
REFERENCE: file containing reference SNPs with two alleles
SUMSTAT.trans: file, where the second allele has been added
```

[11] You might want to replace indels-alleles first e.g. a SNP with alleles T/TCCC into alleles D/I

Please check that the number of SNPs in your output file is expectedly high.

Quick-and-Dirty split SNP name into CHR and POS:

awk '{if (NR==1) {$SNPSUM= "CHR\tPOS"; print }else {split($SNPSUM,a,":"); $1= a[1]"\t"a[2]; print}}'

`SNPSUM: column-number containing snpname in sumstat file`

From this:

MarkerName	Allele1	Allele2	Freq1	FreqSE	Weight	Zscore	P.value	Direction	HetISq	HetChiSq	HetDf	HetPVal	Effect	StdErr
5:29439275	t	c	0.362	0.0045	2098	-0.471	0.6373	+-	75.8	4.139	1	0.04192	-0.057	0.0924
5:106244831:G_GA	i	r	0.3922	6e-04	2098	-1.427	0.1537	--	29.8	1.425	1	0.2326	-0.1216	0.0896`
5:85928892	t	c	0.0635	0.006	2098	1.076	0.2818	++	0	0.014	1	0.9067	0.2003	0.1863

Into this:

CHR	POS	Allele1	Allele2	Freq1	FreqSE	Weight	Zscore	P.value	Direction	HetISq	HetChiSq	HetDf	HetPVal	Effect	StdErr
5	29439275	t	c	0.362	0.0045	2098	-0.471	0.6373	+-	75.8	4.139	1	0.04192	-0.057	0.0924
5	106244831	i	r	0.3922	6e-04	2098	-1.427	0.1537	--	29.8	1.425	1	0.2326	-0.1216	0.0896`
5	85928892	t	c	0.0635	0.006	2098	1.076	0.2818	++	0	0.014	1	0.9067	0.2003	0.1863