Using IMPUTE2 for phasing of GWAS and subsequent imputation

Bryan Howie[α] and Jonathan Marchini[β]

[α]University of Chicago

[β]University of Oxford

23 September 2010

OVERVIEW

In this document, we show how to use IMPUTE2 to impute from a reference panel of phased haplotypes into a genome-wide association dataset (e.g., a set of cases and controls genotyped on a SNP chip). We place particular emphasis on the idea of "pre-phasing" the GWAS data without a reference panel, then using the resulting haplotypes to perform fast imputation from a reference panel of interest. Separating the phasing and imputation steps in this way[1] is beneficial because most of the computing burden of imputation comes from accounting for the unknown phase of the GWAS individuals; if the GWAS data were provided as haplotypes rather than unphased diplotypes, all imputation methods would be much faster. This point is particularly relevant when an investigator wants to perform multiple rounds of imputation into a single GWAS dataset – e.g., re-imputing as larger reference panels become available over time. In that context, it can be wasteful to re-phase the dataset with each round of imputation. This document explains how IMPUTE2 can be used to phase a GWAS dataset and perform fast downstream imputation with the resulting haplotypes.

We begin with a short preamble and quick-start guide, followed by an introduction to the required files and their formats. We then discuss the preferred strategy for applying IMPUTE2 in a genome-wide analysis, which involves splitting the genome into non-overlapping chunks (typically 5 Mb each), phasing/imputing the chunks separately on parallel computer processors, and stitching the imputation results back together into whole-chromosome output files, which can then be passed

[1] "Conventional" imputation algorithms – IMPUTE1, MACH1, BEAGLE, fastPHASE – perform phasing and imputation in a single, analytical step that takes advantage of the structure of their hidden Markov models of DNA sequence variation. By contrast, IMPUTE2 has separate phasing and imputation modules, and it alternates between these during the course of an MCMC run. The IMPUTE2 strategy was motivated both by the computational considerations discussed in this document and by the fact that, when the phasing step is isolated, more of the information in the data can inform the phasing. This leads to more accurate GWAS haplotypes and ultimately to more accurate imputation.

directly to our association analysis program SNPTEST. Within this framework, we demonstrate a couple of ways that IMPUTE2 can be used for imputation (ranging from "best-guess" GWAS haplotypes to full posterior averaging over the GWAS phase uncertainty), and we provide some guidance about the running time vs. accuracy tradeoffs of these strategies.

We also briefly discuss some empirical results and computational benchmarks. On a realistic dataset and a modest computing cluster, IMPUTE2 can perform genome-wide phasing in ~12 hours of real time and genome-wide imputation (given phased haplotypes from the first step) in ~1.5 hours, with a light RAM footprint and high accuracy.

TABLE OF CONTENTS

SOFTWARE WEBPAGE

This document focuses on the basic functionality of IMPUTE2, without getting into the many bells and whistles that are implemented in the program. If you want more details, the most definitive and up-to-date information about the program can be found at our software webpage:
https://mathgen.stats.ox.ac.uk/impute/impute_v2.html.

HOW TO USE THIS MANUAL

This document probably contains more detail than you need to know. In the spirit of "show, don't tell", we have also created some working bash scripts and example files that illustrate how to run IMPUTE2 in the scenarios discussed below. Studying these examples is probably the fastest way to learn the structure of the analysis, so we suggest that you start there. Once you understand the basic workflow, you can use this guide as a definitive reference for specific questions about parameter choices and implementation details.

The QUICK START guide below shows how to perform a simple imputation analysis with the example files. The example scripts are described in more detail in the section on BUILDING A PIPELINE at the end of the document.

QUICK START

The example scripts and data reside in the `impute2_examples/` directory that was included with this download. Note that the copy of IMPUTE2 in this directory was compiled to work on a linux 64-bit computer; if you are running a different architecture, you will need to replace the IMPUTE2 executable with an appropriate one downloaded from our website.

The first thing you need to do is phase your GWAS data. To do so, you can type the following text on the command line (press `Enter` when you're done):

```
./prototype_phasing_job.sh 10 1 1e6
```

This command will invoke a bash script called `prototype_phasing_job.sh`. This script takes three arguments: the chromosome number, the starting chromosomal position of the analysis region, and the ending position of the analysis region. As we explain in some detail later on, the latter two arguments are very helpful for splitting up the genome into tractable pieces that can be analyzed on a parallel computing cluster. In this example, we have asked IMPUTE2 to phase only the first million bases on chromosome 10.

Now that we have phased our GWAS genotypes, we can use the resulting haplotypes to impute additional SNPs from a reference panel. There are two modules for

performing the imputation. The first module uses the "best-guess" haplotypes from the phasing step above:

```
./prototype_imputation_job_best_guess_haps.sh 10 1 1e6
```

Note that the chromosome number and analysis chunk boundaries are the same as in the first run; it is important that these arguments remain the same across the phasing and imputation steps.

Another way of doing the imputation is to use multiple realizations of the GWAS haplotypes, which were sampled during the phasing run, as follows:

```
./prototype_imputation_job_posterior_sampled_haps.sh 10 1 1e6
```

There should now be three files in the `results/` subdirectory that end in the suffix '`.impute2`'. Two of these have the word 'imputation' in their file names – these are your results! We will spend the rest of this document explaining what exactly is happening in each of these scripts and how they fit together.

REQUIRED FILES

To perform an imputation analysis with IMPUTE2, four basic kinds of files are needed:

1. GWAS genotypes file – File containing SNP chip genotypes for a set of GWAS individuals. The format is described at http://www.stats.ox.ac.uk/~marchini/software/gwas/file_format.html#Genotype_File_Format. We distribute a program called GTOOL (http://www.well.ox.ac.uk/~cfreeman/software/gwas/gtool.html) that can convert PED files into this format.

2. Genetic map file – File containing genetic map positions (i.e., fine-scale recombination rates). This file is used to set some parameters of IMPUTE2's hidden Markov model. Genetic map files for all human autosomes can be downloaded from here: https://mathgen.stats.ox.ac.uk/wtccc-software/recombination_rates/genetic_map_b36_combined.tgz.

3. Reference panel haplotypes file – File containing phased reference panel haplotypes for imputation. Each row is a SNP and each column is a haplotype, with columns separated by single spaces. SNPs are assumed to be biallelic, and haplotype alleles must consist entirely of 0's and 1's.

4. Reference panel legend file – File containing information about the SNPs in the haplotypes file, including rsID, chromosomal position, identity of allele labeled '0', and identity of allele labeled '1'. This file contains one line for each line in the haplotypes file, plus a header line. Paired legend and haplotype

files from the 1,000 Genomes Project can be downloaded from here: https://mathgen.stats.ox.ac.uk/impute/1000g_jun2010_b36_ceu.tgz. Other versions of these files (e.g., for different populations) are available on the IMPUTE2 webpage.

IMPUTE2 does not read chromosome numbers from input files (unlike, say, PLINK), so each of these four kinds of files must be split up by chromosome. We have already done this for the genetic map and reference panel files linked above, so you just need to ensure that your formatted GWAS genotypes are divided into one file per chromosome.

If your GWAS genotypes have already been filtered and aligned to the forward strand of the human reference sequence, you will not need any other input files. Otherwise, we provide a number of options for strand alignment and SNP/sample filtering in the software, with details described here: https://mathgen.stats.ox.ac.uk/impute/impute_v2.html#Program_Options.

BASIC WORKFLOW

Once you have obtained the necessary input files for your imputation analysis, you will need to perform the following steps, which are explained further in the sections that follow:

1. Apply standard SNP and sample filters to your GWAS genotypes.

2. Split each chromosome into smaller chunks for analysis. Each chunk will be regarded as a separate, fully parallelizable analysis unit until Step 5.

3. Phase the genotyped SNPs in your GWAS dataset.

    a. Produce best-guess haplotypes for every GWAS individual in a given analysis chunk.

    b. Store multiple versions of the GWAS haplotypes, which are statistically sampled by the phasing algorithm during an MCMC run.

4. Use the best-guess or sampled haplotypes (the choice depends on running time/accuracy considerations) to impute all of the reference-panel-only SNPs in a chunk.

5. Combine chunk-specific imputation results into whole-chromosome imputation output files.

6. Run post-imputation QC and association tests on the output file from each chromosome.

SNP AND SAMPLE FILTERING OF GWAS DATA

Before getting started with phasing/imputation, you should apply standard QC filters to your GWAS data to remove SNPs with genotyping artifacts and individuals with systematically poor genotype quality. It is also helpful to align the allele coding of your genotypes to the forward strand of the human reference sequence, since this will make the imputation step easier.

SPLITTING THE GENOME INTO CHUNKS FOR PARALLEL ANALYSIS

A genome-wide IMPUTE2 analysis usually begins by breaking each chromosome into smaller chunks, typically ~5 Mb each. There are at least three good reasons to do this. First, dividing up the data facilitates parallel computing, which prevents large chromosomes from causing bottlenecks in the analysis pipeline. Second, IMPUTE2 uses a population genetic approximation that works best in regions which have not experienced a large amount of historical recombination, which means that you will get better accuracy in a 5 Mb chunk than on a whole chromosome. Third, splitting up a chromosome should allow you to analyze all of your study individuals together, which will lead to better phasing and imputation accuracy than you would obtain if you subdivided the GWAS individuals for faster computation.

You do not need to physically split your dataset into chunk-specific files; IMPUTE2 has a command-line option called `–int` that will read only the desired region from a whole-chromosome file. For example, typing '`–int 5e6 10e6`' in a call to IMPUTE2 would cause it to process only positions 5,000,000–10,000,000 on the current chromosome.

The main theoretical danger of splitting chromosomes into separate analysis chunks is that accuracy may decrease near the edges of each chunk[2]. To guard against this, IMPUTE2 automatically adds an internal buffer region (default=250 kb) to either side of each chunk. This buffer is used in the phasing/imputation, but it does not appear in the final output file. You can change the buffer size via the command-line argument `–buffer` (details on website).

[2] A related, hypothetical worry about analyzing sub-chromosomal chunks is that rare alleles on long haplotype backgrounds may become harder to impute. One way to mitigate this effect is to use larger buffer regions around each chunk. It is also worth noting that statistical phasing has inherently limited accuracy, so this kind of signal might be obscured by phasing errors even if you analyzed very large chunks.

The `–int` and `–buffer` options make it easy to specify chunk boundaries. For example, you could define the following 5-Mb analysis intervals on a given chromosome:

[1, 5e6]
[5e6+1, 10e6]
[10e6+1, 15e6]
…

Consider the second chunk: if you set '`-int 5000001 10e6 –buffer 250`' on the command line, the true analysis interval would be [5e6+1-0.25e6, 10e6+0.25e6], but the output file would only contain SNPs in [5e6+1, 10e6]. Hence, defining non-overlapping chunks in this way will cause every SNP to appear in exactly one output file, and you can simply concatenate the chunk-specific imputation output files to produce a chromosome-wide output file. Combining phased haplotypes across chunks is possible, if slightly trickier; we will not discuss the topic here since you do not need to combine the haplotypes across chunks to perform imputation from pre-phased haplotypes.

The easiest way to define chunks for a chromosome is as shown above: start at '1' and then apply a constant increment until the end of the chromosome is reached. This strategy generally works, although it may stumble in a few special cases:

1. At either end of a chromosome, the reference panel SNPs will occasionally extend more than a chunk increment (e.g., 5 Mb) past the last SNP in your GWAS data. The program cannot make predictions in chunks with no GWAS SNPs, so the best approach is to merge such chunks with neighboring chunks that do contain GWAS SNPs[3].

2. Chunks that span the centromere may sometimes contain very few SNPs. These can be merged with flanking chunks, if desired. One way to avoid this issue is to define chunk boundaries separately on each arm of a metacentric chromosome.

3. If you start at position '1' and define chunks by a constant increment, the last chunk on a chromosome may contain very few SNPs by chance. We usually merge the last chunk with the second-to-last chunk to avoid poor imputation at the ends of chromosomes.

Once you have defined the chunk boundaries for each chromosome, it is up to you how to store this information and convey it to IMPUTE2. We explain the mechanism that we use in the section on BUILDING A PIPELINE at the end of this document.

[3] Note, however, that the imputation quality will still be low in regions where there are reference panel SNPs but not GWAS SNPs over multi-megabase stretches.

PRE-PHASING GWAS GENOTYPES

Once you have settled on chunk boundaries within each chromosome, the next step is to phase your GWAS data in each chunk and store the results for later. IMPUTE2 uses a Markov chain Monte Carlo (MCMC) algorithm to phase a set of genotypes, and this can produce two kinds of output:

1. Posterior-sampled haplotypes. Each "iteration" of the MCMC generates one realization of the haplotypes underlying your GWAS data, as inferred from a statistical model of population genetic variation. If you want to store the most complete output possible (which will in turn give you the greatest flexibility in downstream analysis), you can ask the program to print out every set of haplotypes that it samples during a phasing run.

2. As an alternative, you can reduce the posterior haplotype samples into a single set of consensus or "best-guess" haplotypes for your GWAS data in a given chunk. This approach will inevitably discard some of the information from the phasing (e.g., about the level and distribution of uncertainty in the haplotypes), but it can simplify and speed up downstream imputation.

These two output settings can be used separately or together. In the prototype analysis pipeline that is implemented in the example scripts provided with this download, we produce both kinds of output at the same time, and we think that this is good practice in general.

Printing posterior haplotype samples

IMPUTE2 includes a few command-line options to facilitate printing of posterior haplotype samples. The `-stage_one` and `—hap_samp_dir` options are designed to work together by automatically naming the sampled haplotype files in a fairly foolproof manner. The `-stage_one` flag tells the program that this is the first step in a two-stage imputation run, where the first stage phases the observed data (printing posterior haplotype samples along the way) and the second stage imputes untyped SNPs using the sampled haplotypes and a reference panel. If your GWAS genotype file is named `[-g]` (which is the IMPUTE2 input option for that file) and your chunk boundaries are [A,B], the `-stage_one` flag will cause the program to print files with names like these:

```
[-g]_posA-B_hapsamp1.gz
[-g]_posA-B_hapsamp2.gz
[-g]_posA-B_hapsamp3.gz
...
```

Each file contains the full set of haplotypes (two per GWAS individual, comprised of all SNPs in the current chunk+buffer) from a single iteration of the MCMC, and the iteration number is reported as `hapsamp[iteration]`. The file names also include the chunk boundaries in the form `posA-B`; this distinguishes the results of the

current analysis from other chunks on the same chromosome that may be running in parallel.

By default, `-stage_one` will cause these sampled haplotype files to be printed in the same directory as your input genotype file. If you would prefer to put them somewhere else, you can specify a different directory as an argument to the `–hap_samp_dir` option. We provide an example of this in our prototype analysis pipeline.

If you would rather have more control over the file naming instead of relying on IMPUTE2's automated system, there is also an alternative option called `–hap_samp_out_g`. If you would like more information about how this works, please check the online documentation or contact the authors.

Printing best-guess haplotypes

If you would like IMPUTE2 to print best-guess haplotypes, you simply need to activate two flags on the command line: `-phase` and `–include_buffer_in_output`. The `-phase` flag tells the program to create and print best-guess haplotypes, and the `–include_buffer_in_output` flag specifies that the haplotypes should include the buffer region. The buffer is usually omitted from IMPUTE2 output files, but in this case we want to preserve it in order to avoid edge effects in the next stage of the analysis (imputation).

When `-phase` is activated, the program will print the best-guess haplotypes to a file called `[-o]_haps`, where `[-o]` is the name of the main output file that you specified on the command line. For those who are technically-inclined: these haplotypes are produced internally from the MCMC by "minimizing the expected switch error" over the posterior haplotype samples.

USING PRE–PHASED GWAS HAPLOTYPES FOR DOWNSTREAM IMPUTATION

Now that you have phased your GWAS genotypes, it should be very fast to impute untyped SNPs from a reference panel of haplotypes. We find that it is easiest to continue analyzing the data in chunks at this stage of the pipeline – i.e., the haplotypes you produced in a given chromosome chunk will now be used to perform imputation in that chunk.

As explained in the section on REQUIRED FILES near the beginning of this document, the basic ingredients of an IMPUTE2 imputation analysis are a GWAS genotypes file, a genetic map file, a reference panel haplotypes file, and a reference panel legend file; the command-line options for these are `–g`, `–m`, `–h` , and `–l`, respectively. The discussion in this section assumes that these files have been provided to the program, with exceptions noted below.

In parallel to the previous section, we can impute the missing genotypes from either posterior haplotype samples or best-guess haplotypes. Imputing from best-guess haplotypes will be faster, but it will also sacrifice some accuracy; imputing from sampled haplotypes is preferable from a statistical point of view (since it accounts for the uncertainty of the GWAS phasing) and yields more accurate results, but this small boost in accuracy may not be worth the additional running time if resources are limited.

Imputing from posterior haplotype samples

Much as the `-stage_one` flag was used to help name sampled haplotype files in the pre-phasing step, the `-stage_two` flag is designed to carry this information through to the imputation step. In fact, you should be able to use almost exactly the same command-line call for this step, except that `-stage_two` will replace `-stage_one`, the `-h` and `-l` reference panel files will be included this time, and we will drop the phasing-specific options (`–phase` and `–include_buffer_in_output`). This can be seen by comparing the IMPUTE2 calls in the example scripts `protype_phasing_job.sh` and `prototype_imputation_job_posterior_sampled_haps.sh`.

By default, IMPUTE2 will read in the sampled haplotypes from each phasing iteration (excluding the initial burn-in iterations) and use them to impute the untyped SNPs; it will then average across these imputations to produce posterior probabilities for each unobserved genotype. Averaging over iterations in this way is good statistical practice, but for *N* iterations (not counting burn-in) the running time will be roughly *N* times longer than it would take to impute from a single set of best-guess haplotypes.

As an in-between option, you can choose to impute from every *j*th set of sampled haplotypes out of *N* possible sets; this is controlled by the `–thin` option in IMPUTE2. By default, `–thin` is equal to 1; i.e., the program imputes from every set of sampled haplotypes. If you instead set '`–thin 5`' on the command line, the program would only impute from the 1st, 6th, 11th,… sets of sampled haplotypes. For a thinning interval of *j*, the running time will be reduced by a factor of *j*, and the accuracy will be somewhat less than if you had averaged across all of the sampled haplotype sets.

Imputing from best-guess haplotypes

Imputing from best-guess haplotypes is very simple. IMPUTE2 will perform a single imputation step rather than running MCMC in this situation, and the main change on the command line is to replace the `–g` file (unphased GWAS genotypes) with a `–known_haps_g` file. This is just the best-guess haplotype output file from the phasing step (the file whose name ends in `_haps`). IMPUTE2 will impute the untyped alleles in each phased haplotype, then convert these allelic probabilities into diploid genotype probabilities for each GWAS individual, which is trivial under the reasonable assumption of HWE.

COMBINING CHUNK-SPECIFIC OUTPUT FILES INTO WHOLE-CHROMOSOME FILES

The steps described above will produce a number of chunk-specific imputation output files for each chromosome, with file names specified by the –o option in IMPUTE2. Before passing this information to SNPTEST (our association testing software), it is usually helpful to combine these output files into full-chromosome files. On linux computing systems, this is simply a matter of concatenating the chunk-specific –o files using the 'cat' command. For example:

```
cat chr12.chunk1.output chr12.chunk2.output ... > \
chr12.allChunks.output
```

Note that the chunk-specific output files should be concatenated in order of their positions along a chromosome for easier downstream interpretation.

PERFORMING POST-IMPUTATION QC AND ASSOCIATION TESTING

Jonathan Marchini distributes an association-testing program called SNPTEST that is designed to work seamlessly with output files from IMPUTE2. SNPTEST performs a number of Bayesian and frequentist association tests, and it also reports metrics that can be used to filter out poorly imputed SNPs. For more details, please visit the SNPTEST webpage:
http://www.stats.ox.ac.uk/~marchini/software/gwas/snptest.html.

EMPIRICAL RESULTS AND COMPUTATIONAL BENCHMARKS

To get a rough idea of the computational burden and accuracy of a pre-phasing+imputation analysis, we performed an experiment using real data from the Wellcome Trust Case Control Consortium (WTCCC). We started with 2,500 controls from the 1958 British Birth Cohort who were genotyped on both the Affymetrix 6.0 and Illumina 1.2 M platforms. We masked all chromosome 10 SNPs except those on the Affymetrix 500k platform, and we phased this pseudo-Affy 500k data using the phasing tools described above. Once we had generated haplotypes for the observed data, we imputed the masked SNPs from a reference panel of 120 CEU haplotypes from the 1,000 Genomes project. By comparing the masked-then-imputed genotypes with their true values, we were able to assess the accuracy of different imputation strategies.

On a 2.5 Ghz processor, it took about two hours and 500-700 Mb of RAM to phase an average 5-Mb chunk on IMPUTE2's default settings (k=80, iter=30, burnin=10); when spread across a computing cluster with 100 processors, it took ~12 hours of real time to phase every chunk in the genome. We then used the best-guess haplotypes from this analysis to impute all of the 1,000 Genomes SNPs that are not typed on the Affy 500k array. This step took about 15 minutes per chunk, or 90 minutes to impute the entire genome on a cluster, with about 100 Mb of RAM

required per process. (Double-check the RAM usage.) We also ran the IMPUTE2 imputation module that uses posterior haplotype samples. The running time for the pre-phasing step was the same as before, but the imputation step took *N* times longer, where *N* is the number of sampled haplotype sets used for the imputation. Most of these computational benchmarks would change linearly with the number of GWAS SNPs and the number of GWAS individuals; the imputation benchmarks would increase linearly with the number of haplotypes in the reference panel.

The imputation results were most accurate when we averaged across 20 sets of sampled GWAS haplotypes, although the best-guess GWAS haplotypes achieved quite good imputation accuracy in much less time. We also ran IMPUTE1 (which does not have a separate phasing step) on this dataset for comparison. Imputation from best-guess haplotypes achieved similar accuracy to IMPUTE1 in ~20x less time (not counting the pre-phasing investment that generated the best-guess haplotypes).

PRACTICAL GUIDANCE

When you want to perform a single imputation analysis, we believe that the generic IMPUTE2 method provides the gold standard for accuracy and is computationally feasible for genome-wide association studies. However, the development of high-throughput DNA sequencing technologies is causing imputation reference data to proliferate quickly, and in this context many people would like to re-impute their GWAS multiple times as the reference panels expand. To save computing costs in this environment, we have developed some modifications to IMPUTE2 can make each round of imputation much faster, conditional on an initial investment in phasing the GWAS genotypes well.

We have covered many of the nuances of using IMPUTE2 in this document, but our basic recommendation is simple: if you are going to pre-phase a GWAS to speed up multiple rounds of imputation, you should produce a set of best-guess GWAS haplotypes and impute from these. Each imputation step will be extremely fast, and while the accuracy will not be as high as possible, it will be fine for most purposes.

If computing resources and expertise allow, we also recommend storing the posterior haplotype samples from the GWAS phasing. Even if these files are not used in most rounds of imputation, keeping them around allows you to run the more intensive and accurate imputation modules in the future without having to re-phase your GWAS data. For example, you might use the best-guess haplotypes to quickly explore the results that arise from new reference panels, and then use the sampled haplotypes to impute from a "final", stable reference panel as you prepare to submit a manuscript.

BUILDING A PIPELINE

In the QUICK START section near the beginning of this document, we did not discuss all of the example scripts in the `impute2_examples/` directory that comes with this download. The additional scripts are meant to provide a template for setting up a fully automated parallel computing pipeline. The scripts may not work "right out of the box", depending on the details of your computing environment, but they should nonetheless give you a good idea of how we run this kind of analysis on a genome-wide scale. Expert users are welcome to do things differently, but we thought it would be helpful to provide a concrete example for people with less experience.

In this discussion, we will focus on the phasing scripts for simplicitly; the analogous imputation scripts follow an almost identical structure. The top-level script is called `master_phasing_script.sh`. This is a simple bash script that calls a client R script for each human autosome (chromosomes 1-22).

The R script (`submit_impute2_jobs_to_cluster.R`) accepts a couple of command-line arguments. One of these specifies which chromosome to process; it is convenient to make a separate call to the script for each chromosome because the data should be divided into chromosome-level files. There are also three boolean arguments. By setting one of these to 'TRUE' on the command line, you tell the script what kind of IMPUTE2 functionality to invoke.

Once it has processed your command-line directives, the R script does two things. First, it reads in a separate file that defines the analysis chunks for the current chromosome. There is an example of our version of this file in the provided directory (`data_files/analysis_chunks_5Mb_chr10.txt`). Next, the script uses the 'qsub' command to submit an appropriate IMPUTE2 job to the grid engine (which is essentially a piece of software that manages a computing cluster). The actual IMPUTE2 command is embedded in one of the bash scripts that we introduced in the QUICK START section. The grid engine will decide which processor to use for the job and get it started – now you just need to wait for the results!

We hope that this example pipeline is helpful, and that you may even be able to use some of these scripts for your own analyses. You will almost certainly need to change parts of these files, and you should make an effort to understand what they're doing by reading this document, but you are welcome to build on the backbone we have provided.

CONTACTING THE AUTHORS

If you have questions about this instruction manual, or about IMPUTE2 in general, you should contact both of the following people:

Dr. Bryan Howie – bhowie@uchicago.edu
Dr. Jonathan Marchini – marchini@stats.ox.ac.uk

If you have specific questions about why the program isn't working properly on your system, please send a copy of the 'summary' output file with your e-mail. Thanks for using IMPUTE, and good luck.