iPat: Interface of Prediction and Association Tools for Genomics

Chun-Peng James Chen1 and Zhiwu Zhang1

1Department of Crop and Soil Sciences, Washington State University, Pullman, WA 99164, USA

# Abstract

The advances of sequencing technologies brought two areas of rapid developments, one is genome wide association studies and the other is prediction of phenotypes from genotypes. Both areas require sophisticated statistical models. Multiple packages were developed to perform such analyses, including TASSEL, PLINK, rrBLUP, GAPIT, GenABEL, MLMM, FarmCPU and BLINK. Most of these packages, except TASSEL, use command line interface (CLI) to realize the optimum efficiency. These packages require substantial of training on DOS command or R language, which eliminating many users who are comfortable only with graphic user interface (GUI).

The software iPat was developed for such need to transform existing CLI packages such as GAPIT, rrBLUP, and FarmCPU, into a graphic user interface without the compromise on efficiency. With iPat, users can perform analyses by simply clicking mousses to link existing data files, select models, and define input parameters. iPat was developed in Java to use the cutting edge technologies for programming across languages such as Java, C and R. Additional benefit include that iPat can be run on any operating systems including Windows and Mac. The executable file of iPat can be freely download from <http://ZZLab.Net/iPat>. The website also contains user manual, tutorials and demonstration dataset.

[Target Journal: Bioinformatics, Software]

Note: Since we planed to implement some additional feature into our software recently, and the detail is still under developed and immature. Currently the content involved “format conversion” and “Computing time” can’t be presented in this paper, but will update soon.

# Introduction

As the rapid development of sequencing technology, both cost and time for collecting genome-level data has been dramatically decreased and shortened. Hence, genome wide association studies (GWAS) has become much more feasible under limited resources. Even so, due to the constrain of computer memory and the risk of statistic error inflation when we perform analysis on such a huge data, an efficient and omnibus software become highly demanded nowadays. An ideal software should be able to implement data management, quality control and estimating parameters in statistic models by well-define algorithms. Fortunately, there are several packages have already been developed to meet such needs. Software like PLINK, rrBLUP, GAPIT and FarmCPU are capable of doing robust analysis without effort, and most of them are still releasing updates to improve either statistical power or computing speed nowadays.

However, not all users are comfortable with these software packages. Given the fact that most of the tools are developed as command-line tools at the first place, which require users to define the analysis by typing several commands into a text-based interface. Those scientists who have limited programming skills will easily be frustrated and lose opportunities to implement more specific analysis. Second, for users who don’t use certain tools frequently, it’s difficult for them to memorize the meanings of each input arguments in abbreviation, e.g. ‘-t’, ‘-m’ in PLINK or ‘group.from’ in GAPIT. One another thing is, no matter what software is used, data formatting error is a pretty common difficulty that users may confront. Not a single format can be globally eligible among different software.

The software iPat was developed to deal with those mentioned drawbacks. This is a software that transform existing command-line interface (CLI) tools such as GAPIT, FarmCPU and PLINK into a graphic user interface (GUI) without the compromise on efficiency. With iPat, users can perform analyses by simply clicking mouse to import data files, select models, and define input parameters. Besides, parameters supported by iPat can implement the majority type of tasks in the original tools, and they are redesigned as a set of compact parameters that users can easily understand their meaning. Users can also hover mouse on the names of each parameter to learn the further definition when they are still ambiguous with those parameters. Last, iPat is compatible with most of different data format, automatically converting the imported files to a suitable format based on which tool is called. In this article, the correctness of the conversion will also be discussed.

# Method

* 1. Software structure

A complete analysis in iPat can be defined as two phases: the configuring phase and the computing phase. In the configuring phase, users are required to import the input data, choose which CLI tool to be used, and define the parameters in the analysis. Based on the imported files and the selected tool, iPat will automatically convert the file format into an eligible one, or will directly enter next phase if the format is just fit to the tool. A converted file will be generated in the same folder as where the original file exists. In the computing phase, the input parameters and file paths will be passed to the CLI tool, and iPat will then create a new thread for the CLI tool to execute the commands. As a GUI, iPat doesn’t do much change from the original tools. The way how iPat communicates with those CLI tools will be detailed discussed in the later paragraph.

The whole GUI was developed in Java, which ensures the optimum runtime speed in performing format conversion and passing input parameters to CLI tools.

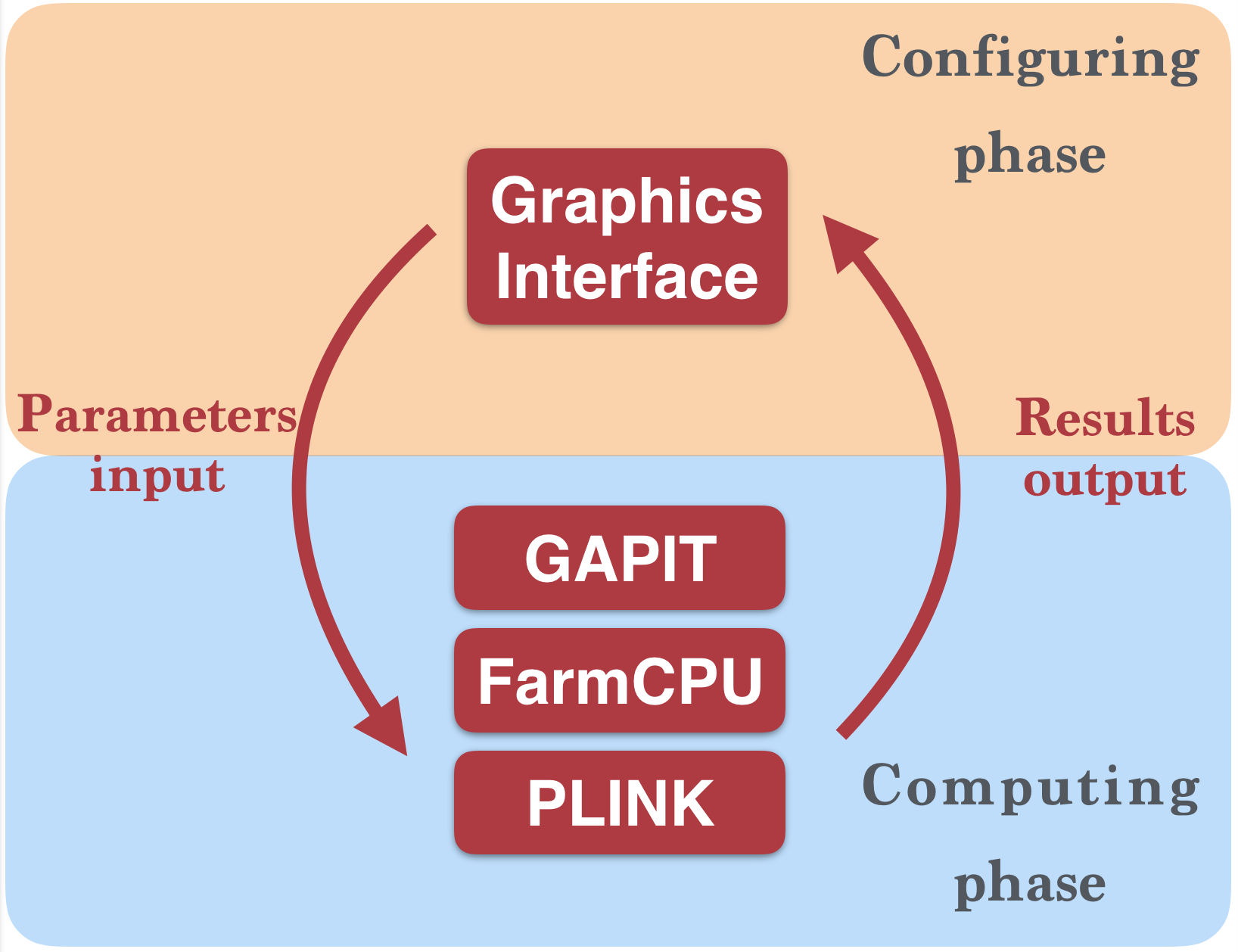


Figure 1. The software structure of iPat

* 1. R-based tools

In order to call R-based CLI tools from iPat, each tool would have a corresponding R script file to tell iPat how to translates users’ commands into R language. The R script file will also install all the prerequisite packages and then call the tool function to start the analysis. Therefore, what iPat does after receiving input commands is to open a new thread and execute this R script file by calling “Rscript” function in the command-line interpreters (i.e. MS-DOS in Windows system, or Terminal in MacOS/Linux system), and the script file will continue to launch the analysis accordingly.

* 1. C-based tools

C driven tools are easier to be launched than R-based tools, since this type of tools can be directly called by the command-line interpreter, rather than be launched in the specific development environment. Once receiving input parameters from users, iPat will call the tool function through the command-line interpreter, and analysis will begin to run based on the specified configuration. However, it requires users to specify the path where the executable file of the tool located at the first time.

* 1. Format conversion

(Pending)

# Result

* 1. Computing speed

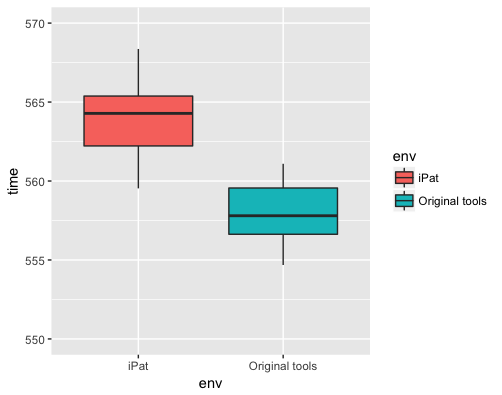


Figure 2. The comparison of runtime speed among different platform.

* 1. Correctness (format conversion)

(Pending)