BIOINFORMATICS APPLICATIONS NOTE Vol. 28 no. 23 2012, pages 3134–3136
doi:10.1093/bioinformatics/bts571

Genome analysis

Advance Access publication September 27, 2012

Advanced Complex Trait Analysis

A. Gray[1,*], I. Stewart[2] and A. Tenesa[2]

[1]EPCC, The University of Edinburgh, Edinburgh EH9 3JZ and [2]The Roslin Institute, The University of Edinburgh, Edinburgh EH25 9RG, UK

Associate Editor: Alex Bateman

ABSTRACT

Motivation: The Genome-wide Complex Trait Analysis (GCTA) software package can quantify the contribution of genetic variation to phenotypic variation for complex traits. However, as those datasets of interest continue to increase in size, GCTA becomes increasingly computationally prohibitive. We present an adapted version, Advanced Complex Trait Analysis (ACTA), demonstrating dramatically improved performance.

Results: We restructure the genetic relationship matrix (GRM) estimation phase of the code and introduce the highly optimized parallel Basic Linear Algebra Subprograms (BLAS) library combined with manual parallelization and optimization. We introduce the Linear Algebra PACKage (LAPACK) library into the restricted maximum likelihood (REML) analysis stage. For a test case with 8999 individuals and 279 435 single nucleotide polymorphisms (SNPs), we reduce the total runtime, using a compute node with two multi-core Intel Nehalem CPUs, from ~17 h to ~11 min.

Availability and implementation: The source code is fully available under the GNU Public License, along with Linux binaries. For more information see http://www.epcc.ed.ac.uk/software-products/acta.

Contact: a.gray@ed.ac.uk

Supplementary information: Supplementary data are available at *Bioinformatics* online.

Received on June 15, 2012; revised on August 22, 2012; accepted on September 17, 2012

1 INTRODUCTION

Complex traits are determined by large numbers of genetic and environmental factors as well as their interactions. Identifying the contributing genes and quantifying their effects in the context of one or multiple environments is of key importance in the development of improved breeding strategies in livestock, the identification of therapeutic targets for animal and human disease, and the understanding of how natural and artificial selection shape the genomes of animals and humans.

Genome-Wide Complex Trait Analysis (GCTA) (Yang *et al.*, 2011) is a freely available C++ software package for quantifying the contribution of genetic variation to phenotypic variation for complex traits. This comprises two stages: estimation of the 'Genetic Relationship Matrix' (GRM), which contains a measure of genetic correlation between individuals and is calculated using single nucleotide polymorphism (SNP) data, and the use of restricted maximum likelihood (REML) to maximize the likelihood of the phenotypes given the GRM. The specific algorithmic implementation in GCTA is sub-optimal when compiled for modern multi-core processors (which are discussed in the Supplementary Material). In this article, we present an adapted version of the code, Advanced Complex Trait Analysis (ACTA), which offers dramatic performance improvements.

In Section 2, we describe the model used by GCTA, and present the adaptations performed to the GRM estimation and REML analysis stages of the code respectively. In Section 3, we evaluate and discuss the effects of these developments.

*To whom correspondence should be addressed.

2 BACKGROUND AND METHODS

We first summarize the mixed linear model (MLM) used by GCTA: for full details see (Yang *et al.*, 2011), noting that for clarity we change notation slightly. In the below, i and j always denote indices over a total of P individuals, and k denotes an index over a total of N SNPs.

GCTA involves the fitting of SNP effects using the MLM

$$\mathbf{y} = \mathbf{X}\beta + \mathbf{W}\mathbf{u} + \epsilon \text{ with } \mathbf{V} = \mathbf{W}\mathbf{W}^{\mathbf{T}}\sigma_u^2 + \mathbf{I}\sigma_\epsilon^2, \quad (1)$$

where y_i is the phenotype of the i_{th} individual and $\mathbf{u}$ is a vector of SNP effects with normal distribution $N(0, \mathbf{I}\sigma_u^2)$, i.e. u_k is the phenotypic effect attributable to the k_{th} SNP. $\mathbf{W}$ is the $P \times N$ *incidence matrix* which projects the SNP effects to $\mathbf{y}$: it contains the genotype for each individual and is given by

$$W_{ik} = \frac{(r_{ik} - 2p_k)}{\sqrt{2p_k(1-p_k)}}. \quad (2)$$

where r_{ik} is the number of copies of the reference allele for the k_{th} SNP of the i_{th} individual, and p_k is the frequency of the reference allele. Similarly, $\mathbf{X}$ is an incidence matrix for the β vector of fixed effects. ϵ is a vector of residual effects with normal distribution $N(0, \mathbf{I}\sigma_\epsilon^2)$. $\mathbf{V}$, the variance of $\mathbf{y}$, is given as a linear combination of the SNP component σ_u^2 and the residual component σ_ϵ^2. $\mathbf{I}$ is the identity matrix. It is useful to define $\mathbf{g} = \mathbf{W}\mathbf{u}$, i.e. g_i is the total genetic effect for the i_{th} individual, and also the GRM $\mathbf{A} = \mathbf{W}\mathbf{W}^{\mathbf{T}}/N$, giving

$$\mathbf{y} = \mathbf{X}\beta + \mathbf{g} + \epsilon \text{ with } \mathbf{V} = \mathbf{A}\sigma_g^2 + \mathbf{I}\sigma_\epsilon^2. \quad (3)$$

where $\sigma_g^2 = N\sigma_u^2$. A_{ij} contains a measure of genetic correlation between the i_{th} and j_{th} individuals, and can be calculated as follows:

$$A_{ij} = \frac{1}{N}\sum_{k=1}^{N} W_{ik}W_{jk}. \quad (4)$$

A main purpose of GCTA is the determination of the σ_g^2 and σ_ϵ^2 variance components through a fit of the observed phenotypic variance to the observed genotypic variance through the MLM. Prior to the fit, the GRM **A** must be calculated, and GCTA provides functionality to output this to file for use in a subsequent variance component determination, or for use externally.

The implementation of Equation 4 is by far the most computationally demanding section of the GRM estimation; the performance of this part dictates the performance of the code as a whole. We first describe the original GCTA implementation, and then go on to present our improvements. Full implementation details are given in the Supplementary Material).

The use of the constant N in Equation 4 assumes that the genotype for each of the P individuals includes information for all N SNPs. However, in reality genotypes often have missing information, resulting in a number of W_{ik} being invalid, and GCTA accommodates this. Each invalid W_{ik} is tagged as such by being assigned a specific value (outside the range of possible valid values). An `if` statement is used, within the innermost loop, to only include valid W values in the determination of each A_{ij}. The constant divisor N is replaced by $\hat{\mathbf{N}}$, where each $\hat{N}_{ij}$ is determined through counter incrementation as the total number of SNPs contributing to a specific A_{ij}.

This implementation is sequential, i.e. it will only make use of a single core in a multi-core system. Furthermore, it does not strive to fully exploit the fast on-chip memory caches or the SIMD vector processing units, utilization of which is vital for efficient use of each core on the CPU. Furthermore, the `if` statement and counter incrementation in the innermost loop use valuable compute cycles and restrict compiler optimization.

In ACTA, we restructure the algorithm into the following distinct stages: we determine each $\hat{N}_{ij}$ semi-analytically; we set each invalid element of W to zero; we perform the matrix multiplication WW^T (including all elements of `W`); and finally we divide each A_{ij} by $\hat{N}_{ij}$. The dominant stage now becomes a clean matrix multiplication, for which we exploit the widely available, parallel and highly optimized Basic Linear Algebra Subprograms (BLAS) library (Dongarra *et al*., 1990), which makes efficient use of each core in a multi-core system. These developments dramatically reduce the time taken for this dominant code section, to the extent that a range of other code sections become important in terms of the time taken to create the GRM. We therefore also introduce manual parallelization via OpenMP directives (to allow utilization of multiple cores) (OpenMP Forum, 2011), plus a range of serial optimizations (for example the restructuring of code sections to remove redundant operations).

Once the GRM **A** has been calculated, it can be used to perform a fit to observed phenotypes through the MLM. In GCTA this is done through REML analysis. Full details are described in (Yang *et al*., 2011); for this article, it is sufficient to know that this process is computationally dominated by the inversion of **V**, which is a function of **A**, and is of size $P \times P$. The original code performs this matrix inversion through Cholesky decomposition using the `LDLT` class of the Eigen C template library. This first decomposes V into the lower triangular matrix L and diagonal matrix D, such that $V = LDL^T$. Subsequently, the matrix V^{-1} is found using the forward substitution technique (again using Eigen functionality).

We replace the use of Eigen with the alternative Linear Algebra PACKage (LAPACK) library (Anderson *et al*., 1999). We convert V from a matrix in Eigen format to a regular array, and pass a pointer to this structure to the LAPACK `dpotrf` Cholesky factorization routine. This performs a similar operation to the above, except the decomposition is to the form $V = LL^T$. The LAPACK `dpotri` routine then operates on this decomposed matrix to give V^{-1}, again through forward substitution. Internally, LAPACK uses calls to BLAS linear algebra operations, again resulting in a parallel and highly efficient implementation.

The algorithms used in ACTA are numerically equivalent to those in GCTA, so any differences in reported results are purely attributable to rounding errors. For the dataset described in the following section, we found the ACTA results to be fully consistent with those from GCTA: numerical differences in the reported variance component estimates were seen to be at least two orders of magnitude lower than the reported standard error on the value.

3 RESULTS AND DISCUSSION

The total time taken by the code is problem specific, depending on the number of individuals, SNPs and REML iterations needed to converge. We profile the original GCTA code on a compute node comprising two Intel Nehalem 4-core CPUs, using the *SLVC* test case with 8999 individuals and 279 435 SNPs distributed throughout all of the autosomal chromosomes. We find that the total time for the analysis is just under 17 h: 94% on the GRM estimation and the remaining 6% on the REML analysis (which requires five iterations to converge). The version in use is GCTA 0.92.2.

In Figure 1, we compare this performance to the new ACTA code, where we use BLAS and LAPACK from the Intel Math Kernel Library. For ACTA, we vary the number of OpenMP threads utilized from 1 (i.e. utilizing just a single core) to 8 (fully utilizing the resource). We decompose timings into the GRM and REML stages. The performance advantage, for the GRM estimation, of ACTA over GCTA is a factor of 21.4 when using a single thread (owing to more efficient utilization of the core as

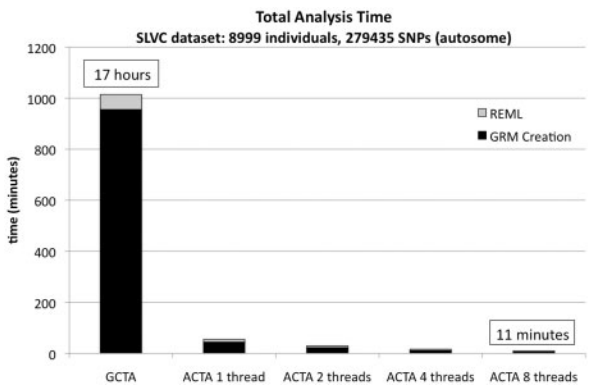

Fig. 1. The total compute time for a case with 8999 individuals and 279 435 SNPs. ACTA is compared with the original GCTA package. The effect of increasing the number of parallel threads utilized in ACTA is shown. The timings are decomposed into the GRM estimation and REML stages

discussed in the previous section and Supplementary Material); this rises to a factor of 151.7 when fully utilizing the resource with 8 threads. The equivalent factors for the REML stage are 5.4 (1 thread) and 13.7 (8 threads). The combined time for the test case is reduced from ~17 h to ~11 min. We also measure the REML analysis time for the same case using the ASReml software (Gilmour *et al*., 2006) to be just under 5.5 h: this is a factor of 76 times slower than ACTA (and a factor of 5.6 times slower than GCTA). For discussion on the implementational differences between GCTA and ASReml see (Yang *et al*., 2011).

Our adapted software, which we call ACTA is timely not least because of the imminent arrival of sequencing data involving large numbers of people and SNPs. There is an increase in the number of re-sequencing projects and the imputation of large numbers of GWAS samples to the one thousand genomes project. There is an urgent need for efficient computational tools, and ACTA builds on the functionality offered by GCTA while dramatically improving performance and hence the feasibility of what can be practically achieved.

Funding: This work was supported by Cancer Research UK [C12229/A13154].

Conflict of Interest: none declared.

REFERENCES

Anderson,E. *et al.* (1999) *LAPACK Users' Guide*. 3rd edn. Society for Industrial and Applied Mathematics, Philadelphia, PA.

Dongarra,J. *et al.* (1990) A set of Level 3 Basic Linear Algebra Subprograms. *ACM Trans. Math. Soft*., **16**, 1–17.

Gilmour,A.R. *et al.* (2006) *ASReml User Guide. Release 2.0*. VSN International Ltd, Hemel Hempstead, UK.

OpenMP Forum (2011) The OpenMP API specification for Parallel Program Version 3.1.

Yang,J. *et al.* (2011) GCTA: a tool for genome-wide complex trait analysis. *Am. J. Hum. Genet*., **88**, 76–82.