CoGAPS computational costs

The dominant computational cost of CoGAPS is the MCMC matrix factorization performed by the GAPS-JAGS C++ package. In this supplement, we specify the memory requirements and processing time of this GAPS matrix factorization. All of the examples provided in this supplement were conducted on a Dell PowerEdge1950 dual-Xeon 64-bit workstation, with 4Gb of RAM and running RedHat Enterprise 64-bit Linux. Throughout this supplement, N refers to the number of genes, M the number of conditions, and n_p the number of patterns.

Memory usage

Throughout the implementation of the GAPS matrix factorization, GAPS-JAGS stores $\mathbf{D}$, $\mathbf{\Sigma}$, and the atomic distributions for $\mathbf{A}$ and $\mathbf{P}$. As required by the JAGS infrastructure, GAPS-JAGS also stores the values of $\mathbf{A}$ and $\mathbf{P}$ from the previous iteration and the new value for either the $\mathbf{A}$ or $\mathbf{P}$ matrix that is being updated during the current iteration. Generally, the number of atoms in the atomic distribution will be proportional to the number of elements in corresponding matrix. Although the constant of proportionality cannot be known a priori, we will assume for the sake of the memory usage presented here that the number of atoms approximately equals the number of matrix elements. Table 1 outlines the resulting memory usage for each of these variables. From this table, it is apparent that the dominant memory usage in GAPS-JAGS is the storage of $\mathbf{D}$ and $\mathbf{\Sigma}$.

Table 1: Memory usage of variables stored in GAPS-JAGS

Variable	Data Type	Number of Elements
$\mathbf{D}$	double	$N \times M$
$\mathbf{\Sigma}$	double	$N \times M$
$\mathbf{A}$	double	$N \times n_p$
$\mathbf{P}$	double	$n_p \times M$
$\mathbf{A}$ atom locations	long long int	$N \times n_p$
$\mathbf{A}$ atom masses	double	$N \times n_p$
$\mathbf{P}$ atom locations	long long int	$n_p \times M$
$\mathbf{P}$ atom masses	double	$n_p \times M$

Processing time

At each iteration, GAPS updates an atom or pair of atoms for the $\mathbf{A}$ or $\mathbf{P}$ matrix, requiring a corresponding maximum computational cost of $\mathrm{O}(N \log N)$ or $\mathrm{O}(M \log M)$ to sort the atoms in the domain. The algorithm then involves an $\mathrm{O}(M)$ or $\mathrm{O}(N)$ calculation of the change in likelihood due to this update in the corresponding **A** or **P** matrices, respectively. Therefore, the dominant computational cost of the GAPS matrix factorization algorithm is $\mathrm{O}(N \log N)$. Table 2 includes the processing time of factoring varying subsets of the genes in the [1] 1363 gene by 9 conditions GIST data set into 5 patterns over 10000000 burn-in and 5000000 chain iterations.

Table 2: Processing time for GAPS matrix factorization of the [1] GIST data set.

Number of genes	Total processing time (s)	Processing time per iteration (s)
10	1723	1.1×10^{-5}
50	2478	1.7×10^{-5}
100	3306	2.2×10^{-5}
500	9831	6.6×10^{-5}
1000	17583	1.2×10^{-4}

Figure 1 shows the dependence of the processing time per iteration on the number of genes. From this figure, it is apparent that the $\mathrm{O}(N)$ processing time dominates for the number of genes considered in this application.

References

[1] M.F. Ochs, L. Rink, C. Tarn, S. Mburu, T. Taguchi, B. Eisenberg, and A.K. Godwin. Detection of treatment-induced changes in signaling pathways in gastrointestinal stromal tumors using transcriptomic data. Cancer Res, 69(23):9125–9132, 2009.

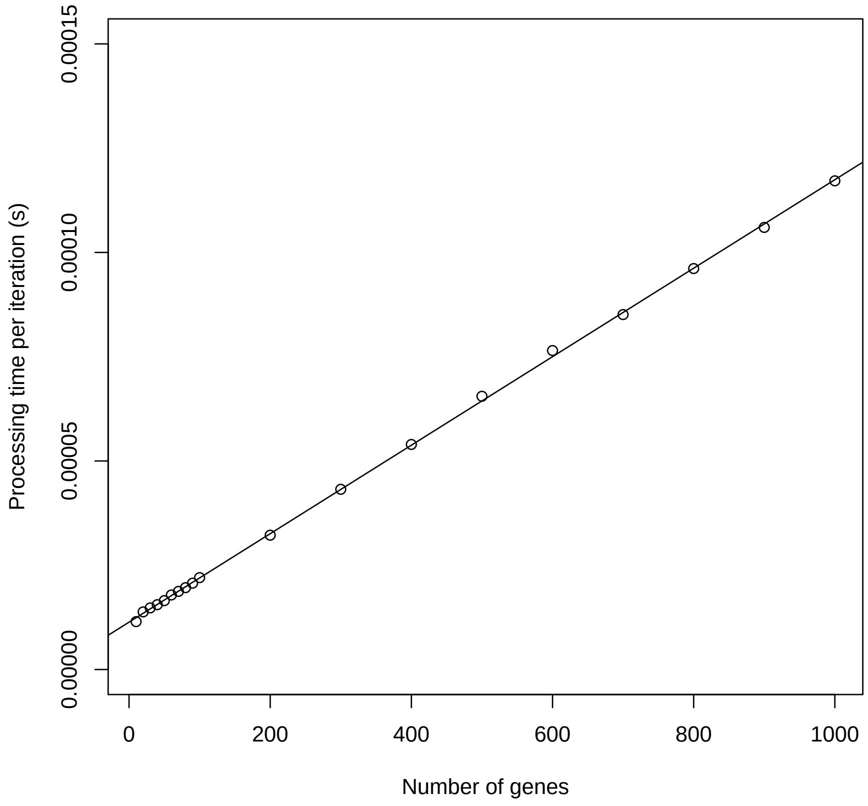

Figure 1: Processing time per iteration for varying number of genes.