A weighted network has every gene connected to every other, but as such, correlation networks are becoming the cornerstone of gene co-expression networks (GCN). These GCN’s can be drawn graphically, such that every gene is represented as a spot (or node) and pairs of genes are connected together with a line (or edge). A weighted network has every gene connected to every other, but the thickness (or weight) of the edge is determined by the strength of the correlation between the gene pairs. An unweighted network has edges between a pair of genes if, and only if, the correlation between the gene pairs is above a pre-defined threshold. GCN analysis is a systems biology approach whereby clusters (modules) of highly co-expressed (or connected) genes are identified and used to explain roles of genes for which no previous biological role has been found. GCN has enabled researchers to make reasonable predictions about the functions of genes for which no previous biological role has been found. As such, correlation networks are becoming the cornerstone of gene co-expression networks (GCN). These GCN’s can be drawn graphically, such that every gene is represented as a spot (or node) and pairs of genes are connected together with a line (or edge). A weighted network has every gene connected to every other, but the thickness (or weight) of the edge is determined by the strength of the correlation between the gene pairs. An unweighted network has edges between a pair of genes if, and only if, the correlation between the gene pairs is above a pre-defined threshold. GCN analysis is a systems biology approach whereby clusters (modules) of highly co-expressed (or connected) genes are identified and used to explain patterns seen across experimental conditions.

For any trio of genes (A, B and C), if there is a strong correlation between AC and BC, it follows that there is likely to be a strong correlation between AB (Figure 1). This confounding of direct and indirect associations leads to a spurious edge forming between AB and is likely to cause problems when it comes to identifying and interpreting gene modules. It has been shown that the distribution of the Pearson correlation coefficient (r) varies for different numbers of experimental conditions (Reverter and Chan, 2008). This has implications for choosing an appropriate threshold for r in reconstructing a weighted network. To help address these issues, the PCIT algorithm (Reverter and Chan, 2008) was developed. This algorithm uses first-order partial correlation coefficients combined with an information theory approach to identify meaningful gene–gene associations. Both approaches have previously been applied independently in reconstructing gene networks (Basso et al., 2005; de la Fuente et al., 2004; Magwene and Kim, 2004). PCIT is a completely data-driven approach, thus removing the need for choosing an arbitrary and possibly subjective correlation threshold. While the original PCIT algorithm has been successfully applied to a series of medium sized networks (>5000 genes) (Hudson et al., 2009), its application to larger networks (>10 000 genes) is limited due to time required to perform calculations on all possible trios of genes. For organisms with relatively fewer total transcripts such as Plasmodium falciparum (4300 transcripts represented on an Affymetrix GeneChip®) this is not an issue as all transcripts can be analysed with standard computing resources. However, higher eukaryotes such as drosophila, bovine, arabisdopsis, rat and human, have more transcripts (18 500–47 000 transcripts represented on an Affymetrix GeneChip®) making the task of applying PCIT to even half the total number of transcripts unfeasible due to time constraints (Figure 2a).

We make the PCIT algorithm publicly available to researchers, using the R programming language through this PCIT R package. The package has the ability to make use of parallel computing to greatly improve the speed of the PCIT algorithm through the use of Rmpi (http://cran.r-project.org/package=Rmpi), an R package wrapper to the Message Passing Interface (MPI). In the absence of a suitable parallel computing environment a serial implementation, where calculations are executed sequentially, is the default.

**ABSTRACT**

**Summary:** We make the PCIT algorithm, used for detecting meaningful gene–gene associations in co-expression networks, available as an R package. Automatic detection of a suitable parallel environment is used such that scripts are portable between parallel and non-parallel environments with no modification of the script.

**Availability and implementation:** Source code and binaries freely available (under GPL-3) for download via CRAN at http://cran.r-project.org/package=PCIT, implemented in R and supported on Linux and MS Windows.

**Contact:** nathan.watson-haigh@csiro.au

1 INTRODUCTION

Gene expression profiling has led to the identification of genes that perform in a coordinated manner. This guilt-by-association heuristic has enabled researchers to make reasonable predictions about the roles of genes for which no previous biological role has been found. As such, correlation networks are becoming the cornerstone of gene co-expression networks (GCN). These GCN’s can be drawn graphically, such that every gene is represented as a spot (or node) and pairs of genes are connected together with a line (or edge). A weighted network has every gene connected to every other, but the thickness (or weight) of the edge is determined by the strength of the correlation between the gene pairs. An unweighted network has edges between a pair of genes if, and only if, the correlation between the gene pairs is above a pre-defined threshold. GCN analysis is a systems biology approach whereby clusters (modules) of highly co-expressed (or connected) genes are identified and used to explain patterns seen across experimental conditions.

For any trio of genes (A, B and C), if there is a strong correlation between AC and BC, it follows that there is likely to be a strong correlation between AB (Figure 1). This confounding of direct and indirect associations leads to a spurious edge forming between AB and is likely to cause problems when it comes to identifying and interpreting gene modules. It has been shown that the distribution of the Pearson correlation coefficient (r) varies for different numbers of experimental conditions (Reverter and Chan, 2008). This has implications for choosing an appropriate threshold for r in reconstructing a weighted network. To help address these issues, the PCIT algorithm (Reverter and Chan, 2008) was developed. This algorithm uses first-order partial correlation coefficients combined with an information theory approach to identify meaningful gene–gene associations. Both approaches have previously been applied independently in reconstructing gene networks (Basso et al., 2005; de la Fuente et al., 2004; Magwene and Kim, 2004). PCIT is a completely data-driven approach, thus removing the need for choosing an arbitrary and possibly subjective correlation threshold. While the original PCIT algorithm has been successfully applied to a series of medium sized networks (>5000 genes) (Hudson et al., 2009), its application to larger networks (>10 000 genes) is limited due to time required to perform calculations on all possible trios of genes. For organisms with relatively fewer total transcripts such as Plasmodium falciparum (4300 transcripts represented on an Affymetrix GeneChip®) this is not an issue as all transcripts can be analysed with standard computing resources. However, higher eukaryotes such as drosophila, bovine, arabisdopsis, rat and human, have more transcripts (18 500–47 000 transcripts represented on an Affymetrix GeneChip®) making the task of applying PCIT to even half the total number of transcripts unfeasible due to time constraints (Figure 2a).

We make the PCIT algorithm publicly available to researchers, using the R programming language through this PCIT R package. The package has the ability to make use of parallel computing to greatly improve the speed of the PCIT algorithm through the use of Rmpi (http://cran.r-project.org/package=Rmpi), an R package wrapper to the Message Passing Interface (MPI). In the absence of a suitable parallel computing environment a serial implementation, where calculations are executed sequentially, is the default.
Fig. 2. PCIT performance. (a) graph showing the speed of the serial implementation with project times for data sets containing 16 000 and 32 000 genes. (b) the speed of the parallel implementation of PCIT as a percentage of the time taken for the serial implementation.

3 METHODS

The PCIT algorithm calculates first-order partial correlation coefficients for every trio of genes. The total number of gene trio combinations is given by

\[ \binom{n}{3} = \frac{n!}{3!(n-3)!}, \]

where \( n \) is the total number of genes and \( d = 3 \). The algorithm has cubic complexity (doubling the number of genes increases the time required to run PCIT by 8-fold) due to the fact that calculations are performed on \( \binom{n}{3} \) gene trios and so does not scale well (Figure 2a). This together with the memory requirements for performing operations on large, dense correlation matrices means that even modern desktop computers will be unable to perform the PCIT algorithm on more than several thousand (~8000) genes. However, the set of calculations performed for each trio of genes are entirely independent and are ideal candidates for execution in parallel. Workload is distributed evenly among the slaves in order to maximise speed by assigning sets of gene trios to a task. The task is then carried out by a slave CPU. This assignment of gene trios into sets is based on which gene is in the first position in a trio and is the total number of genes and the gene constrained in the first position.

\[ \sum_{m=1}^{n-2} \binom{m}{2} \leq \binom{n}{3}, \]

where \( n \) is the total number of genes and the gene constrained in the first position.

3.1 Memory requirements

The PCIT algorithm has cubic complexity and as such performing PCIT on a correlation matrix of 16 000 genes (approximately half and a third of the rat and human transcripts represented on Affymetrix GeneChips®, respectively) would take almost 6 days to complete (Figure 2a). The parallel implementation of the PCIT algorithm provides substantial speed-ups for around 4–5 CPUs (Figure 2b). The use of additional CPUs (>5) provides smaller increases in speed which plateau at around eight CPUs. Larger data sets are able to achieve more substantial speed-ups using the same number of CPUs. However, a plateau of ~10–15% appears to be forming as a maximum achievable speed-up.

4 QUICK START GUIDE

Firstly, install the latest version of the R programming environment and then install the PCIT package:

> install.packages("PCIT")

If you are using an MPI parallel programming environment, ensure you install and configure the Rmpi package:

> install.packages("Rmpi")

If you are using MS Windows on a system with multiple processing cores and wish to utilise them for parallel processing, we advise you to follow instructions provided by the Department of Statistical & Actuarial Sciences at the University of Western Ontario: http://www.stats.uwo.ca/faculty/yu/Rmpi/ for running Rmpi under DeinoMPI as this is the simplest for utilising standalone dual and quad-core MS Windows machines under MPI.

Once R is started you may view documentation, run demos and load example data using standard R functions:

# View information about the PCIT package
> help(package="PCIT")

# View help for the pctl() function
> ?pctl
The function of applying the PCIT algorithm is `pcit()`. In its simplest form, it only requires a correlation matrix as input and returns a result from which one can obtain the correlation matrix indices for those correlations deemed to be significant.

```r
# Perform PCIT on the correlation matrix
result <- pcit(c)
# Get indices for the meaningful correlations
signif <- idx(result)
# Plot the distribution of meaningful correlations
plotCorCoeff(c, list("PCIT Meaningful" = + signif), col=c("red"))
```

Since the PCIT algorithm is not restricted to GCNs but can be applied to any correlation-based network, we have not provided functions for handling microarray data. Instead we expect the user to appropriately handle, normalize and filter data, whatever its origin, and provide the correlation matrix for use in PCIT. However, we provide here a brief example for handling microarray data in the form of an ExpressionSet object from Bioconductor (Gentleman et al., 2004):

```r
# Install the Biobase package from Bioconductor
> source("http://bioconductor.org/biocLite.R")
> biocLite("Biobase")
# Load the Biobase package
> library(Biobase)
# Get/create an ExpressionSet object from somewhere: we assume eSet is this object
> dIm(eSet)
exprs Features 15 Samples 1000
# Calculate a correlation matrix of the transposed expression data
c <- cor(t(exprs(eSet)));
# The amount of RAM required for running PCIT can be approximated
> pcitMemoryRequirement(nrow(c), units="MB")
RAM
[1] 22.88818
units
[1] "MB"
# The maximum correlation matrix size (number of rows and columns) which can be handled
> maxMatrixSize(ram=1, units="GB")
[1] 6688
```

The function of applying the PCIT algorithm is `pcit()`. In its simplest form, it only requires a correlation matrix as input and returns a result from which one can obtain the correlation matrix indices for those correlations deemed to be significant.

5 CONCLUSION

The PCIT R package not only provides a flexible means for applying the PCIT algorithm to GCN, but also allows this to be done in parallel in <15% of the time for data sets with ≥8000 genes. This will now allow researchers to apply PCIT to networks containing more genes than had previously been possible within the same amount of time. The use of automatic detection of a suitable parallel (MPI) environment means that scripts are portable between parallel and non-parallel systems in a transparent form and without the need for modification. These features of the PCIT R package, which is available via CRAN, make it attractive to use over other similar packages designed for building GCN (Basso et al., 2005).

ACKNOWLEDGEMENTS

The authors would like to thank the staff of the CSIRO Advanced Scientific Computing Facility for their help and support in setting up and making available the parallel computing environment used during this work. The authors thank the reviewers for insightful comments and suggestions which have contributed to the improvement of this article and provided ideas for future development.

Funding: Commonwealth Scientific and Industrial Research Organisation’s (CSIRO) Office of the Chief Executive (OCE) Postdoctoral Fellowship program and the CSIRO’s Transformational Biology Capability Platform.

Conflict of Interest: none declared.

REFERENCES


