

FastGCN (1.1) Manual

Meimei Liang

October 26, 2014

Institute of Bioinformatics, Zhejiang University.

Changes from version 1.0

Added function of imputation by overall mean of all expression values.

Added function of imputation by overall mean of each gene expression values.

Added function of modules identification.

1.Introduction

FastGCN is a convenient and efficient tool for fast gene co-expression networks. First genetic entropy is exploited to filter out some genes with no or small expression change. Then Pearson Correlation Coefficients are calculated. At last these coefficients are normalized. All these calculating procedures are implemented on GPU. To save the space the coefficient matrix is compressed. To increase the convenience z-score or FDR (False Discovery Rate) is taken as the threshold for output. We packed GPU version, multi-core CPU version, single-thread CPU version and single-thread R version in the download package.

2.Citing FastGCN

If you use FastGCN in any published work, please cite:

Meimei Liang, Futao Zhang, Gulei Jin and Jun Zhu* (2014) FastGCN: a GPU Accelerated Tool for Fast Gene Co-expression Networks.

3.Bugs report

If you have any questions, bug reports or comments with our software, please contact:

Dr. Liang with her email address lmml@zju.edu.cn

or

Prof. Zhu with his email address: jzhu@zju.edu.cn

4.Input Data format

The input file is a white-space (space or tab) delimited file. Currently the input files must contain the individual names in the first row and the gene names in the first column. From this version we can analyze the data with missing values. The missing value should be marked as “NA” in the data files.

gene	s0	s1	s2	s3	s4	s5	s6	s7	s8	s9	s10	s11	s12	s13	s14	s15
PDCL3	0.537333333	0.185333333	0.130833333	0.501333333	0.072833333	0.48										
CFHR5	0.3075	0.199	0.685333333	1.1415	0.4804	0.676166667	0.35825									
OR2K2	1.047	0.35	0.1705	0.2115	1.004	1.2475	0.231	0.8425								
SUMO1	0.2691	0.2113	0.4013	1.03	0.3723	0.8887	0.5918	0.1385								
MMP7	4.275	4.59425	5.3955	4.1975	5.19725	5.48	6.12775	4.23425								
CRNKL1	1.335375	0.84475	0.7565	0.897875	0.309	0.5415	0.701125									
RTN1	0.642625	0.875375	2.769875	1.317125	1.193428571	1.55										
TOB2	0.328833333	0.590166667	0.528583333	0.72825	1.33	1.287916667										
BANK1	0.86275	2.46625	2.36425	3.09475	0.821	2.75175	3.005	4.2845								

5. Getting started with FastGCN

5.1 Download

The source codes and binaries of FastGCN are now available for free download at <http://ibi.zju.edu.cn/software/FastGCN>.

The source code for GPU implementation was written with CUDA and C/C++. The source code for multi-core CPU implementation was written with C/C++ and OpenMP. The source code for single-thread CPU implementation was written with C/C++. And the single-thread R implementation is an R script.

We only offered binaries for Windows platforms and Red Hat Enterprise Linux 6.5. So for Linux/Unix users we strongly recommend downloading the source code and compiling.

5.2 Compilation

5.2.1 GPU implementation

To compile this source code, you should download and install CUDA SDK first from <https://developer.nvidia.com/cuda-toolkit-archive>.

For Windows System, we recommend use a C/C++ IDE such as Microsoft Visual Studio or Eclipse combined with NVIDIA NSight (<http://www.nvidia.com/object/nsight.html>) as the integrated compile environment.

For Linux System, we offered 2 Makefile files in the source code package, one for CUDA 5.5 and one for CUDA 6.0. Unzip the source package in the CUDA SAMPLES directory, then go to the directory where the Makefile is located and type command “make”.

5.2.2 Multi-core CPU implementation

For windows System, you need a C/C++ compiler. But make sure to keep Open MP support open.

For Linux System, you have two options. One is to use the Makefile in the source package, another is to go to the source directory and type the command below:

```
g++ -fopenmp -m64 -o FastGCN_omp FastGCN.cpp DataBlock.cpp pear_cpu.cpp
```

5.2.3 Single-thread C implementation

For windows System, you need a C/C++ compiler. For Linux System, please use the Makefile in the source package or go to the source directory and type:

```
g++ -m64 -o FastGCN_omp FastGCN.cpp DataBlock.cpp pear_cpu.cpp
```

5.3 Run

If using GPU implementation, Multi-core CPU implementation or Single-thread C/C++ implementation, you should input 2, 3, 4, 5 or 6 arguments. For GPU implementation, CUDA driver from NVIDIA should be installed. And make sure that you system can find where the CUDA runtime dynamic link library is located. So put the library in the same directory as the executable binary file or set the “PATH” environmental variable.

./FastGCN_gpu para0 para1 para2 para3 para4 para5

para0: The first parameter is the input file name. This parameter is mandatory.

Para1: The second is the output file name. This parameter is mandatory.

Para2: The 3rd is the threshold of z-score for output, default as 3.29. This parameter is optional.

Para3: The 4th is the parameter for data preprocessing, default as 1. That means all the raw data will be kept. If the parameter is a float number falling into an open interval $(0,1)$, that means the proportion will be kept. If the parameter is an integer number falling into an open interval $(1, \infty)$, that means that number of genes will be kept. This parameter is optional.

Para4: The 5th is the maximum output count number, default as 100 per gene. That means the first maximum count number per gene to be output. Maybe too many co-expressions associated with one gene. For the sake of conveniences,

this tool can take maximum output count number per gene as the parameter.
This parameter is optional.

Para5: The 6th is FDR control parameter, default as being disabled. That means using z-score threshold. When this parameter is enabled and set a value, the 3rd parameter is disabled. This parameter is optional.

The running example is as below:

```
D:\FastGCN>FastGCN_gpu.exe eData_g2000.txt out.txt 3.29 1 100 0.05
*****
FastGCN (a GPU accelerated tool for Fast Gene Co-expression Networks) 1.1
Copyright (c) 2010-2014 by Meimei Liang, Futao Zhang, Gulei Jin and Jun Zh
u
Institute of Bioinformatics, Zhejiang University, China
First use Information Entropy to pretreat and cut off some genes.
Then calculate the Pairwise Correlation Coefficients.
Finally do Z-normalization of the coefficients
You should input 2, 3, 4, 5 or 6 arguments, the first one is the input fil
e name
and the second is the output file name! The 3rd, 4th, 5th 6th are optional
.
The 3rd is the threshold of z-score for output, default as 3.29 .
The 4th is the parameter for data preprocessing, default as 1.
That means all the raw data will be kept.
If the parameter is a float number falling into an open interval (0,1),
that means the proportion will be kept.
If the parameter is a integer number falling into an open interval (1,i
nfinite),
that means that number of genes will be kept.
The 5th is the maximum output count number, default as 100 per gene.
That means the first maximum count number per gene to be output.
The 6th is FDR control parameter, default as being disabled.
That means using z-score threshold.
When this parameter is enabled and set a value, the 3rd parameter is disab
led.
*****
Using GPU?
Device0:NUS 5200M with 1024MB total memory.
Gene Number after cut-off:2000,column Numer:590
Input Process: time:1201
Computing Process: time:983
Module Identification : time:249
Output Process: time:5382
D:\FastGCN>
```

Because R implementation was designed as the comparison version, currently we can't take any parameters. If you use this implementation to analysis data, currently you should modify the R source code to set the parameters, the input file name and the output file name. After finishing this, you can run it with R GUI or Rscript.

The running example is as below:

```
D:\FastGCN>Rscript FastGCN.R
Time difference of 5.210409 secs
D:\FastGCN>dir
```

5.4 Output file

Each row of the output file is a co-expression which generally contains the first gene name, the second gene name, the coefficient, the z-score, the P-value and the Q-value. For GPU implementation, Multi-core CPU implementation or Single-thread C/C++ implementation, if you take z-score as the output threshold, P-values and Q-values would not be calculated. So the output file only contains the first gene names, the second gene names, the coefficients and the z-scores. Because of the input parameter 4, the output file may contain one significant pair two times. We call it a dual output model as showing below:

Gene1	Gene2	z-score	coeff	p-value	q-value
HIGD1B	TEK	3.736641	0.557115	0.000093	0.010112
TEK	HIGD1B	3.736641	0.557115	0.000093	0.010112

For R implementation we output all the information including the first gene name, the second gene name, the coefficient, the z-score, the P-value and the Q-value. And output all the significant pairs within the threshold.

The output file generated by GPU implementation with test data (test_data.txt) is as below (Running Command: FastGCN_gpu.exe test_data.txt out.txt 3.29 1 100 0.05) :

Gene1	Gene2	z-score	coeff	p-value	q-value
BANK1	SELL	4.314422	0.623768	0.000008	0.002625
BANK1	CCL19	3.611350	0.542662	0.000152	0.012378
BANK1	CD160	3.289139	0.505492	0.000503	0.027222
CXCL13	SELL	4.075617	0.596220	0.000023	0.003734
HIGD1B	TEK	3.736641	0.557115	0.000093	0.010112
TEK	HIGD1B	3.736641	0.557115	0.000093	0.010112
CD160	BANK1	3.289139	0.505492	0.000503	0.027222
SELL	BANK1	4.314422	0.623768	0.000008	0.002625
SELL	CXCL13	4.075617	0.596220	0.000023	0.003734
SELL	CCL19	3.117817	0.485728	0.000911	0.042300
CCL19	BANK1	3.611350	0.542662	0.000152	0.012378
CCL19	SELL	3.117817	0.485728	0.000911	0.042300
RMND5B	FSIP1	3.439584	0.522847	0.000291	0.018940
FSIP1	RMND5B	3.439584	0.522847	0.000291	0.018940

The output file generated by R implementation with test data (test_data.txt) is as below:

	gene1	gene2	coeff_pcc	z-score	p_value	q_value
1	TEK	HIGD1B	0.557115	3.736632	9.33E-05	0.010102
2	CD160	BANK1	0.505493	3.289137	0.000502	0.027217
3	SELL	BANK1	0.623767	4.314401	8.00E-06	0.002601

4	SELL	CXCL13	0.596219	4.075601	2.29E-05	0.003729
5	CCL19	BANK1	0.542665	3.61137	0.000152	0.012374
6	CCL19	SELL	0.485729	3.117819	0.000911	0.042295
7	FSIP1	RMND5B	0.522847	3.439577	0.000291	0.018935