SCLS計算機システム講習会

遺伝子ネットワーク並列解析プログラム「BENIGN」実習

大阪大学　大学院情報科学研究科
松田秀雄

2015/09/29

要旨

BENIGNは、複数の生体組織や実験条件下で採取された細胞内の遺伝子の発現データから、各条件下での遺伝子間の発現の依存関係を表すネットワークをベイジアンネットワークにより推定するソフトウェアです。条件ごとに得られたネットワークを相互に比較することで、例えば、時間とともに動的に変化する遺伝子間の依存関係や、細胞組織ごとの遺伝子の働きの違いなどを推定して可視化することができます。

BENIGNの開発では、遺伝子ネットワークの推定部分はSiGN-BN をベースにしています。また、複数の遺伝子ネットワーク推定プロセスの並列実行機能の実現にはMPIDP（GHOST-MP のパッケージに含まれています）をベースにしています。

SiGN-BN: http://www.scls.riken.jp/scruise/software/sign-bn.html

GHOST-MP: http://www.scls.riken.jp/scruise/software/GHOST-MP.html

BENIGNの実行

実行に必要なファイルの用意

BENIGNの実行には、各種パラメータを記載したJobリストファイル、遺伝子発現データを記載したEDFファイルが必要です。なお、これらファイルの詳細については、[付録]ページを参照してください。

ここではあらかじめ用意してあるファイルを元にBENIGNの実行手順を説明します。

実行に必要なファイルの用意

はじめに、BENIGNの実行に必要なファイルを置くディレクトリを作成します。次にファイルをコピーします。

```
$ mkdir sample
$ cd sample
$ cp /home/matsuda/benign/sample/adipo_stage* .
$ ls
adipo_stage1.edf  adipo_stage3.edf  adipo_stages.list
adipo_stage2.edf  adipo_stage4.edf  adipo_stages.sh
```

実行に必要なファイルの用意

コピーするファイルは以下の通りです。

- Jobリストファイル
  - adipo_stages.list
- 遺伝子発現データファイル
  - adipo_stage1.edf
  - adipo_stage2.edf
  - adipo_stage3.edf
  - adipo_stage4.edf
- バッチスクリプト
  - adipo_stages.sh

実行に必要なファイルの用意

使用するJobリストファイルです。

```
$ cat adipo_stages.list
TITLE=adipo_stages
benign -y --blocks 20 -o adipo_stage1 -N 1000 -L 1 -T 0.1 -S linear adipo_stage1.edf
benign -y --blocks 20 -o adipo_stage2 -N 1000 -L 1 -T 0.1 -S linear adipo_stage2.edf
benign -y --blocks 20 -o adipo_stage3 -N 1000 -L 1 -T 0.1 -S linear adipo_stage3.edf
benign -y --blocks 20 -o adipo_stage4 -N 1000 -L 1 -T 0.1 -S linear adipo_stage4.edf
```

TITLE行のadipo_stagesは実行ログファイルに記録されます。

以降のbenignで始まる行は遺伝子ネットワーク推定処理の引数を指定します（行ごとに並列に実行されます）。

-y ダイナミックベイジアンネットワークモデル（時系列データの時に使用できます）。このオプションを省略したときはベイジアンネットワークモデルが選択されます。

--blocks *n* 時系列データの時点数を疑似的に*n*倍に増やす（-yを指定した時は推奨）

-o *name* 出力結果のファイル名の接頭辞を*name*にセット

-N *num* ブートストラップサンプリングを*num*回実行

-L 1 実行履歴を個別のファイルに出力（省略すると標準エラー出力にまとめられる）

-T 0.1 ブートストラップ確率が0.1以上の制御辺のみを出力

-S linear 遺伝子発現量を線形モデルで近似

詳細はSiGN-BNのマニュアル(http://sign.hgc.jp/signbn/manual.html)を参照して下さい。

実行に必要なファイルの用意

使用する遺伝子発現データファイル(抜粋)です。

$ cat adipo_stage1.edf

$Version	1.0					
@PrimaryKeyGroupID	1	1	1	2	2	2
@SecondaryKeyGroupID	1	2	3	1	2	3
Cebpa	-0.141795	-0.138312	-0.194432	-0.176802	-0.155757	-0.079616
Cebpb	-0.222724	-0.126105	0.074613	0.17421	0.233539	0.239671

- $Version はEDFファイルのバージョン情報
- @PrimaryKeyGroupID,@SecondaryKeyGroupIDは実験サンプルID (n=3)
- Cebpa,Cebpbはマウスの遺伝子名（ヒトのC/EBPα, C/EBPβ）

実行に必要なファイルの用意

使用するバッチスクリプトです。

```
$ cat adipo_stages.sh
#!/bin/sh -x
#
#PJM -L "rscgrp=small"
#PJM -L "node=5"
#PJM -L "elapse=20:00"
#

BENIGN=/home/matsuda/benign/bin/benign

export PARALLEL=8
export OMP_NUM_THREADS=$PARALLEL

mpiexec ${BENIGN} -tb adipo_stages.list -lg adipo_stages.benign.log
```

- リソースグループはsmall
- 要求するノードは5ノード
- 実行時間の最大は20分
- スレッド数は8

- 京コンピュータで実行するときはファイルのステージングの指示を加える必要があります
- **Webで公開されているバイナリファイルを使用する**ときは、実行に先立って、chmod +x benignで実行可能権限のパーミッションを設定しておく必要があります

ジョブの投入

ファイルをコピーした後、pjsubコマンドを使いジョブを投入します。

```
$ pjsub adipo_stages.sh
[INFO] PJM 0000 pjsub Job XXXXX submitted.
# XXXXX にはジョブIDが入ります
```

ジョブの投入

実行中のジョブの状況は pjstat コマンドで確認できます。

```
$ pjstat

  ACCEPT QUEUED  STGIN  READY RUNING RUNOUT STGOUT   HOLD  ERROR   TOTAL
       0      0      0      0      1      0      0      0      0       1
s      0      0      0      0      1      0      0      0      0       1

JOB_ID     JOB_NAME   MD ST  USER     START_DATE       ELAPSE_LIM NODE_REQUIRE
XXXXX      adipo_stag NM RUN matsuda  08/26 16:10:02  0000:20:00 5
```

ジョブの投入

ジョブの完了後、以下のファイルが生成されます。

- 実行結果
  - adipo_stage1
  - adipo_stage2
  - adipo_stage3
  - adipo_stage4
- 解析処理のログ
  - adipo_stage1.log.000000
  - adipo_stage2.log.000000
  - adipo_stage3.log.000000
  - adipo_stage4.log.000000

ジョブの投入

ジョブの完了後、以下のファイルが生成されます。

- 実行ログ
  - adipo_stages.benign.log
- 標準出力ファイル
  - adipo_stages.sh.oXXXXX
- 標準エラー出力ファイル
  - adipo_stages.sh.eXXXXX

実行結果

実行結果ファイル(SGN3形式)は次のようになります。

```
$ cat adipo_stage1
SiGN SGN3 FORMAT
[Information]
Edge Attr BS.Prob:double      edgeScore:double    BS.Gain:double      EdgeType:int
          BS.up/down:string   BS.EdgeTypeRatio:string
Node Attr hubness:int         BS.HubIndex:double
X:BS.total:int      1000
[Nodes]
Cebpa     Cebpa     0         1         0.110000
Cebpb     Cebpb     1         8         4.853000
Cebpd     Cebpd     2         3         1.425000
Creb1     Creb1     3         1         0.199000
Egr2      Egr2      4         4         1.331000
Klf15     Klf15     5         1         0.144000
Klf2      Klf2      6         7         3.092000
Klf4      Klf4      7         6         3.650000
Klf5      Klf5      8         1         0.199000
Nr1h3     Nr1h3     9         5         3.513000
Nr3c1     Nr3c1     10        2         1.141000
Pparg     Pparg     11        0         0.000000
Stat5a    Stat5a    12        2         1.722000
Thra      Thra      13        3         0.855000

[Edges]
1         0         0.927000  0.927000  40.873586 0         up        1.00/0.00/0.00
3         0         0.199000  0.199000  21.401210 0         up        1.00/0.00/0.00
8         0         0.199000  0.199000  13.059605 0         up        1.00/0.00/0.00
12        0         0.726000  0.726000  24.049626 1         down      0.00/1.00/0.00
...
```

実行結果

最初は、[Information]セクションです。

```
[Information]
Edge Attr      BS.Prob:double edgeScore:double      BS.Gain:double
       EdgeType:int   BS.up/down:string     BS.EdgeTypeRatio:string
Node Attr      hubness:int    BS.HubIndex:double
X:BS.total:int 1000
```

- [Nodes]、[Edges]セクションが持つ属性のリストを出力します。

実行結果

次に、[Nodes]セクションです。

```
[Nodes]
Cebpa   Cebpa   0       1       0.110000
Cebpb   Cebpb   1       8       4.853000
Cebpd   Cebpd   2       3       1.425000
Creb1   Creb1   3       1       0.199000
Egr2    Egr2    4       4       1.331000
Klf15   Klf15   5       1       0.144000
Klf2    Klf2    6       7       3.092000
Klf4    Klf4    7       6       3.650000
Klf5    Klf5    8       1       0.199000
```

- ノードに関する情報を出力
  - ノード名
  - ノード番号
  - 子ノードの数
  - ブートストラップでのHubIndex

実行結果

最後に、[Edges]セクションです。

```
[Edges]
1       0       0.927000 0.927000 40.873586       0       up      1.00/0.00/0.00
3       0       0.199000 0.199000 21.401210       0       up      1.00/0.00/0.00
8       0       0.199000 0.199000 13.059605       0       up      1.00/0.00/0.00
12      0       0.726000 0.726000 24.049626       1       down    0.00/1.00/0.00
```

エッジに関する情報を出力

- 親ノード番号
- 子ノード番号
- ブートストラップ確率
- 制御の符号（up: 促進、down: 抑制）

など

実行結果

sgn2tsvコマンドで実行結果ファイル(SGN3形式)をTSV形式に変換することができます。

先ほどのジョブの実行結果をsgn2tsvコマンドに渡すと次のようになります。

```
$ /home/matsuda/benign/bin/sgn2tsv adipo_stage1
Wrote edge info to adipo_stage1.edge.tsv
```

実行結果

生成されたTSVファイルは次のようになります。

```
$ cat adipo_stage1.edge.tsv
```

From	To	BinaryRegulation	Regulation	up/down/unknown	BS.Prob	edgeScore	BS.Gain
Cebpb	Cebpa	0	up	1.00/0.00/0.00	0.927	0.927	40.873586
Creb1	Cebpa	0	up	1.00/0.00/0.00	0.199	0.199	21.40121
Klf5	Cebpa	0	up	1.00/0.00/0.00	0.199	0.199	13.059605
Stat5a	Cebpa	1	down	0.00/1.00/0.00	0.726	0.726	24.049626
Egr2	Cebpb	0	up	1.00/0.00/0.00	0.283	0.283	6.670069

- 1行目はヘッダー
- 2行目以降はデータ
- 出力されるデータ
  - 接続元遺伝子
  - 接続先遺伝子
  - Regulation
  - Bootstrap Probability

  など

実行結果の解析

実行結果ファイルを元に、ネットワークの比較、および、
描画を行うことができます。

signprocによるネットワークの比較

signprocコマンドを使ってネットワークを比較することができます。

```
$ /home/matsuda/benign/bin/signproc ¥
--read type=sgn3,file=./adipo_stage1 ¥
--comp type=sgn3,file=./adipo_stage2
```

実行結果は次のページのようになります。

signprocコマンドは、 SiGN-BNのユーティリティプログラムです。
signprocコマンドの詳細については、[付録]および以下のサイトを参照してください。
URL: http://sign.hgc.jp/signproc.html

```
SiGN : signproc  ver. 0.19.0 (Thu Aug 21 11:45:08 2014 JST) (SVN rev. 1526)
  Copyright (C) 2012-2014  SiGN Project Members.

Filter #1 -- --read
Reading a SGN3 network file...
  Edge Attributes: BS.Prob (double) edgeScore (double) BS.Gain (double) EdgeType
(int) BS.up/down (string) BS.EdgeTypeRatio (string)
  Node Attributes: hubness (int) BS.HubIndex (double)
  In total 14 nodes and 44 edges added.

Object: Network
  14 nodes, 44 edges.

Filter #2 -- --comp
CompFilter: Reading... adipo_stage2
Reading a SGN3 network file...
  Edge Attributes: BS.Prob (double) edgeScore (double) BS.Gain (double) EdgeType
(int) BS.up/down (string) BS.EdgeTypeRatio (string)
  Node Attributes: hubness (int) BS.HubIndex (double)
  In total 14 nodes and 43 edges added.
CompFilter: 0 nodes are not found in the given file (2nd network).
COML:    TP       RTP      FP       FN       TN       Sn       Sp
COMP:    14       5        19       22       31       0.389    0.368

Object: Network
  14 nodes, 44 edges.

Finished all the filters.
```

Cytoscapeによるネットワークの描画

実行結果ファイルを元に、CytoscapeやGephi等を使って遺伝子ネットワークを図示することができます。

ここではCytoscape 3でのグラフの描画手順について説明します。

Cytoscapeによるネットワークの描画

まず、実行結果ファイル(SGN3形式)をsgn2tsvコマンドに渡しTSVファイルに変換します。

```
$ /home/matsuda/benign/bin/sgn2tsv ./adipo_stage1
Wrote edge info to adipo_stage1.edge.tsv
```

sgn2tsvコマンドの実行によりエッジ情報をまとめたTSVファイルが生成されます。

```
$ ls adipo_stage1.edge.tsv
adipo_stage1.edge.tsv
```

Cytoscapeによるネットワークの描画

生成されたTSVファイルを次の手順でCytoscapeに読み込みます。

1. Cytoscape を実行する
2. File メニュー → Import → Network → File... を選択する
3. 生成されたTSVファイルを選択する
4. Interaction Definition において
    1. Source InteractionにColumn 1(From)を選択する
    2. Interaction に Column 4(Regulation)を選択する
    3. Target InteractionにColumn 2(To)を選択する
    4. 追加情報として以下の情報を選択する
        1. BS.Prob
        2. BS.Gain
5. 選択後、OKボタンを押す

Cytoscapeによるネットワークの描画

ファイルを読み込んだあと、Layout メニュー からレイアウトを選択します。

1. Control PanelからStyleを選択する
2. 下部にある Edge タブを選択する
3. Target Arrow Shape プロパティを選択する
    1. Column に interaction を選択する
    2. Mapping Type に Discrete Mapping を選択する
    3. down に “T” タイプの矢印を選択する
    4. up に “Arrow” タイプの矢印を選択する
4. Layout メニュー → yFiles Layouts → Organic を選択する

付録

benignコマンド

benignコマンドの書式は以下の通りです。

```
$ benign [-lg log] -tb input [user options]
```

指定可能なオプションについては以下の通りです。

- -lg log
  - 任意
  - BENIGNの実行ログを指定します。
  - 指定がない場合、 mpidp.log という名前でログを出力します。
- -tb input
  - 必須
  - Jobリストファイルを指定します。
- user options
  - -lg, -tb以外のオプションは解析処理へのパラメータとして渡されます。

Jobリストファイルフォーマット

BENIGNのJobリストファイルのフォーマットは以下の通りです。

[TITLE=タイトル]
パラメータ行1
パラメータ行2
…

Jobリストファイルフォーマット

TITLE行の指定は任意であり、指定した場合、実行ログに記録されます。
パラメータ行には、空白区切りでパラメータを指定します。

以下はJobリストファイルの例です。

```
TITLE=example
benign -y --blocks 20 -o ex1 -N 1000 -L 1 -T 0.1 -S linear gene_exp1.edf
benign -y --blocks 20 -o ex2 -N 1000 -L 1 -T 0.1 -S linear gene_exp2.edf
benign -y --blocks 20 -o ex3 -N 1000 -L 1 -T 0.1 -S linear gene_exp3.edf
```

EDFファイルフォーマット

EDFは遺伝子発現データを表現するために設計されたフォーマットです。

タブまたはカンマ区切りのテキストファイルで、次の3つのパーツから構成されます。

1. メタデータセクション
2. 属性セクション
3. データセクション

EDFファイルフォーマット

以下はタブ区切りのEDFの例です。

```
# Meta Data Section
$Version  1.0
# Attribute Section
@PrimaryKeyGroupID      1       1       2       2       3       3
@SecondaryKeyGroupID    1       2       1       2       1       2
# Data Section
gene1                   1.1     2.2     3.3     4.4     5.5     6.6
gene2                   7.7     8.8     9.9     10.1    11.11   12.12
gene3                   13.13   14.14   15.15   16.16   17.17   18.18
```

EDFファイルフォーマット

メタデータセクションでは、メタデータは"$"から始まり、データセット、属性セクションにある属性についてのグローバルな情報を定義します。

属性セクションでは、属性は"@"から始まり、発現サンプルの属性を定義します。

最初のカラムは属性のキーを表します。上の例では、2つの属性キーPrimaryKeyGroupID と SecondaryKeyGroupID が発現サンプルのために指定されています。

SecondaryKeyGroupIDは反復実験の回数を表すのに使います。上の例では発現データでn=2の反復(replicate)で取得した場合を表しています。

データセクションにおいて、各行は遺伝子の発現データを表します。最初のカラムに遺伝子の名前を指定し、以降のカラムに発現データを設定します。

sgn2tsvコマンド

sgn2tsvコマンドはSGN3形式のファイルからTSVファイルを生成するためのプログラムです。
生成したTSVファイルはCytoscape、Gephi等グラフ描画ソフトウェアの入力データとして使用できます。
sgn2tsvファイルの書式は以下の通りです。

```
$ sgn2tsv [-a] <SGN3ファイル>
```

sgn2tsvファイルにSGN3形式のファイルを渡すと、カレントディレクトリ下に<SGN3ファイル>.edge.tsv ファイルが生成されます。-a オプションを指定した場合、<SGN3ファイル>.node.tsv ファイルも生成されます。

生成されるファイルはそれぞれ以下の通りです。

- <SGN3ファイル>.edge.tsv
  - エッジ情報をまとめたTSVファイル
- <SGN3ファイル>.node.tsv
  - ノード情報をまとめたTSVファイル

signprocコマンド

signprocは遺伝子ネットワークファイルに様々な処理(ファイルフォーマットの変換、サブネットワークの抽出、指定したノードの色づけ等)を行うコマンドラインツールです。

signprocはSiGN-BNのユーティリティプログラムです。

ユーザは複数のフィルターを指定することができ、一つのフィルターは一つのネットワークを処理します。

ここでは Comp フィルターについて説明しますが、他のフィルターについては下記URLを参照してください。

http://sign.hgc.jp/signproc.html

signprocのCompフィルター

Compフィルターは2つのネットワークを比較し、結果を標準エラー出力に出力します。

コマンド例

```
$ signproc --read type=sgn3,file=./adipo_stage1 --comp type=sgn3,file=./adipo_stage2
```

オプションは以下の通りです。

- --read type=sgn3,file=./adipo_stage1
  - 最初に読み込むネットワーク
  - type=sgn3
    - ファイルフォーマット
  - file=./adipo_stage1
    - 読み込むファイル
- --comp type=sgn3,file=./adipo_stage2
  - 比較するネットワーク(真のネットワーク)
  - type=sgn3
    - ファイルフォーマット
  - file=./adipo_stage2
    - 読み込むファイル

signprocのCompフィルター

COML, COMP行が比較結果です。
各値は、--comp オプションで指定したadipo_stage2を真のネットワークとした場合の adipo_stage1 の枝を以下の種類ごとに数えた値です。

- TP (True Positive)
  - 両方に存在する
- FP (False Positive)
  - adipo_stage2に存在しないが、adipo_stage1に存在する
- FN (False Negative)
  - adipo_stage2に存在するが、adipo_stage1に存在しない
- TN (True Negative)
  - 両方に存在しない

比較結果の例

	TP	RTP	FP	FN	TN	Sn	Sp
COML:							
COMP:	14	5	19	22	31	0.389	0.368