A pedagogical walkthrough of computational modeling and simulation of Wnt signaling pathway using static causal models in Matlab‡

Shriprakash Sinha*α

Insight, Innovation and Integration
Simulation study involving computational experiments dealing with Wnt signaling pathways abound in literature but often lack a pedagogical perspective that might ease the understanding of beginner students and researchers in transition who intend to work on modeling of the pathway. This paucity might happen due to restrictive policies which enforce an unwanted embargo on the sharing of important scientific knowledge. The manuscript elucidates embedding of prior biological knowledge, integration of heterogeneous information, transformation of biological hypothesis into computational framework and design of experiments in a simple manner interleaved with aspects of Bayesian Network toolbox and Matlab code so as to help readers get a feel of a project related to modeling of the pathway.

Abstract
A tutorial introduction to computational modeling of Wnt signaling pathway in a human colorectal cancer dataset using static Bayesian network models is provided. The walkthrough might aid bio-logists/informaticians in understanding the design of computational experiments that is interleaved with exposition of the Matlab code and causal models from Bayesian Network toolbox. This is done in order to ease the understanding of beginner students and researchers in transition to computational signaling biology, who intend to work in the field of modeling of signaling pathways. The manuscript expounds the computational flow of the contents in advance article¹ via code development and takes the reader in a step by step process of how • the collection and the transformation of the available biological information from literature is done, • the integration of the heterogeneous data and prior biological knowledge in the network is achieved, • conditional probability tables for nodes in biologically inspired tables are estimated, • the simulation study is designed, • the hypothesis regarding a biological phenomena is transformed into computational framework, and • results and inferences drawn using d-connectivity/separability are reported. The manuscript finally ends with a programming assignment to help the readers get hands on experience of a perturbation project. Matlab code with dataset is made available under GNU GPL v3 license at google code project on https://code.google.com/p/static-bn-for-wnt-signaling-pathway.

1 A journey of thousand miles begins with a single step
A tutorial introduction to computational modeling of Wnt signaling pathway in a human colorectal cancer dataset using static Bayesian network models is provided. This work endeavours to expound in detail the simulation study in Matlab along with the code while explaining the concepts related to Bayesian networks. This is done in order to ease the understanding of beginner students and researchers in transition to computational signaling biology, who intend to work in the field of modeling of signaling pathways. The manuscript elucidates embedding of prior biological knowledge, integration of heterogeneous information, transformation of biological hypothesis into computational framework and design of experiments in a simple manner interleaved with aspects of Bayesian Network toolbox and Matlab code so as to help readers get a feel of a project related to modeling of the pathway. Programming along with the exposition in the manuscript could clear up issues faced during the execution of the project.

This manuscript uses the contents of the advance article Sinha¹ as a basis to explain the workflow of a computational simulation project involving Wnt signaling pathway in human colorectal cancer. The aim of Sinha¹ was to computationally test whether the activation of β-catenin and TCF4 based transcription complex always corresponds to the tumorous state of the test sample or not. To achieve this the gene expression data provided by Jiang et al.³ was used in the computational experiments. Further, to refine the model, prior biological knowledge related to the intra/extracellular factors of

‡ This manuscript is a part of the Netherlands Bioinformatics Centre (NBIC) BioRange-II project BR2.5 (code - IGD71G). It complements and uses material from Sinha¹. Netherlands Bioinformatics Centre was supported by the Netherlands Genomics Initiative (NGI) via the Netherlands Genomics Institute in The Netherlands.

alpha Address - 104-Madhurisha Heights Phase 1, Risali Sector, Bhilai - 490006, C.G., India

* Shriprakash Sinha drafted the manuscript. E-mail: shriprakash.sinha@gmail.com

¹–25 | 1
the pathway (available in literature) was integrated along with epigenetic information.

Theory and programming code will be explained in an interleaved manner to help the readers get an insight into the hurdles faced while executing such a project. Material from Sinha\(^1\) will be presented in grey colored boxes and used to explain the various aspects of the Matlab code presented here. Code will be presented in typewriter font and functions in the text will be presented in sans serif.

## 2 Modeling and simulation

### 2.1 Data collection and estimation

An important component of this project is the Bayesian Network Toolbox provided by Murphy \textit{et al.}\(^15\) and made freely available for download on [https://code.google.com/p/bnt/](https://code.google.com/p/bnt/) as well as a Matlab license. Instructions for installations are provided on the website. One can make a directory titled temp with a subdirectory named data and transfer the geneExpression.mat file into data.

```matlab
>> mkdir temp
>> cd temp
>> mkdir data
>>
```

The .mat file contains expression profiles from Jiang \textit{et al.}\(^3\) for genes that play a role in Wnt signaling pathway at an intra/extracellular level and are known to have inhibitory affect on the Wnt pathway due to epigenetic factors. For each of the 24 normal mucosa and 24 human colorectal tumor cases, gene expression values were recorded for 14 genes belonging to the family of SFRP, DKK, WIF1 and DACT. Also, expression values of established Wnt pathway target genes like LEF1, MYC, CD44 and CCND1 were recorded per sample.

The directory temp also contains some of the .m files, parts of contents of which will be explained in the order of execution of the project. The main code begins with a script titled twoHoldOutExp.m. This script contains the function twoHoldOutExp which takes two arguments named evidDence and model. evidDence implies the evidence regarding 'ge' for gene evidence, 'me' for methylation, 'ge+me' for both gene and methylation, while model implies the network model that will be used for simulation. Sinha\(^1\) uses three different models i.e 't1' or \(M_{PBK+EI}\) that contains prior biological knowledge as well as epigenetic information, 't2' or \(M_{PBK}\)

### Table 1 Canonical Wnt Pathway from Sinha\(^1\)

**Canonical Wnt signaling pathway** The canonical Wnt signaling pathway is a transduction mechanism that contributes to embryo development and controls homeostatic self renewal in several tissues (Clevers\(^4\)). Somatic mutations in the pathway are known to be associated with cancer in different parts of the human body. Prominent among them is the colorectal cancer case (Gregorieff and Clevers\(^5\)). In a succinct overview, the Wnt signaling pathway works when the Wnt ligand gets attached to the Frizzled\((fzd)\)/LRP coreceptor complex. \(fzd\) may interact with the Dishevelled \((Dvl)\) causing phosphorylation. It is also thought that Wnts cause phosphorylation of the \(LRP\) via casein kinase 1 (\(CK1\)) and kinase \(GSK3\). These developments further lead to attraction of Axin which causes inhibition of the formation of the degradation complex. The degradation complex constitutes of Axin, the \(\beta\)-catenin transportation complex \(APC, CK1\) and \(GSK3\). When the pathway is active the dissolution of the degradation complex leads to stabilization in the concentration of \(\beta\)-catenin in the cytoplasm. As \(\beta\)-catenin enters into the nucleus it displaces the Groucho and binds with transcription cell factor \(TCF\) thus instigating transcription of Wnt target genes. Groucho acts as lock on \(TCF\) and prevents the transcription of target genes which may induce cancer. In cases when the Wnt ligands are not captured by the coreceptor at the cell membrane, Axin helps in formation of the degradation complex. The degradation complex phosphorylates \(\beta\)-catenin which is then recognized by Fbox/WD repeat protein \(\beta-\text{TrCP}\). \(\beta-\text{TrCP}\) is a component of ubiquitin ligase complex that helps in ubiquitination of \(\beta\)-catenin thus marking it for degradation via the proteasome. Cartoons depicting the phenomena of Wnt activation are shown in figures 1(A) and 1(B), respectively.

**Fig. 1** A cartoon of wnt signaling pathway contributed by Verhaegh et al.\(^2\). Part (A) represents the destruction of \(\beta\)-catenin leading to the inactivation of the wnt target gene. Part (B) represents activation of wnt target gene.
Epigenetic Factors One of the widely studied epigenetic factors is methylation (Costello and Plass\textsuperscript{6}, Das and Singhal\textsuperscript{7}, Issa\textsuperscript{8}). Its occurrence leads to decrease in the gene expression which affects the working of Wnt signaling pathways. Such characteristic trends of gene silencing like that of secreted frizzled-related proteins (SFRP) family in nearly all human colorectal tumor samples have been found at extracellular level (Suzuki et al.\textsuperscript{9}). Similarly, methylation of genes in the Dickkopf (DKK) Niehrs\textsuperscript{10}, Sato et al.\textsuperscript{11}) Dapper antagonist of catenin (DACTx Jiang et al.\textsuperscript{3}) and Wnt inhibitory factor-1 (WIF1 Taniguchi et al.\textsuperscript{12}) family are known to have significant effect on the Wnt pathway. Also, histone modifications (a class of proteins that help in the formation of chromatin which packs the DNA in a special form Strahl and Allis\textsuperscript{13}) can affect gene expression (Peterson et al.\textsuperscript{14}). In the context of the Wnt signaling pathway it has been found that DACT gene family show a peculiar behavior in colorectal cancer (Jiang et al.\textsuperscript{3}). DACT1 and DACT2 showed repression in tumor samples due to increased methylation while DACT3 did not show obvious changes to the interventions. It is indicated that DACT3 promoter is simultaneously modified by both the repressive and activating (bivalent) histone modifications (Jiang et al.\textsuperscript{3}).

Table 2 Epigenetic Factors from Sinha\textsuperscript{1}

that contains only prior biological knowledge and finally, ‘p1’ or $M_{NB+MPBK}$ that is a modified version of naive bayes framework from Verhaegh et al.\textsuperscript{2}. In Matlab, one can type the following

\begin{verbatim}
>> twoHoldOutExp("ge", "t1")
\end{verbatim}

The code begins with the extraction of data from the gene expression matrix by reading the geneExpression.mat file via the function readCustomFile in the readCustomFile.m and generates the following variables as the output - (1) uniqueGenes - name of genes gleaned from the file, (2) expressionMatrix - 2D matrix containing the gene expression per sample data (3) noGenes - total number of genes available (4) noSamples - total number of samples available (5) groundTruthLabels - original labels available from the files (6) transGroundTruthLabels - labels transformed into numerals.

\begin{verbatim}
% Data Collection
% ======
% Extract data from the gene expression
% matrix
[uniqueGenes, expressionMatrix,...
noGenes,noSamples,groundTruthLabels,...
transGroundTruthLabels] = ...
readCustomFile('data/geneExpression.mat');
\end{verbatim}

2.2 Assumed and estimated probabilities from literature

Next, the probability values for some of the nodes in the network is loaded, depending on the type of the network. Why these assumed and estimated probabilities have been addressed in the beginning of the computation experiment will be explained later. Meanwhile, the estimation of probabilities is achieved through the function called dataStorage in the dataStorage.m. The function takes the name of the model as an input argument and returns the name of the file called probabilities.mat in the variable filename. The mat file contains all the assumed and computed probabilities of nodes for which data is available and is loaded into the workspace of the Matlab for further use.

\begin{verbatim}
% Load probability values for some of
% the nodes in the network
fname = dataStorage(model);
load(fname);
\end{verbatim}

\begin{enumerate}
\item Repressive Histone Mark $H3K27me3$ for DACT3 11 Loci from Jiang et al.\textsuperscript{3} were adopted. Via fold enrichment, the affects of the $H3K27me3$ was found 500 bp downstream of and near the DACT3 transcription start site (TSS) in HT29 cells. These marks were recorded via chromatin immuno-precipitation (ChIP) assays and enriched at 11 different loci in the 3.5 kb to 3.5 kb region of the DACT3 TSS. Fold enrichment measurements of $H3K27me3$ for normal FHs74Int and cancerous SW480 were recorded and normalized. The final probabilities are the average of the normalized values.

\item Active Histone Mark $H3K4me3$ for DACT3 Loci from Jiang et al.\textsuperscript{3} were adopted. Via fold enrichment, the affects of the $H3K4me3$ was found 500 bp downstream of and near the DACT3 transcription start site (TSS) in HT29 cells. These marks were recorded via chromatin immuno-precipitation (ChIP) assays and enriched at 11 different loci in the 3.5 kb to 3.5 kb region of the DACT3 TSS. Fold enrichment measurements of $H3K4me3$ for normal FHs74Int and cancerous SW480 were recorded and normalized. The final probabilities are the average of the normalized values.

\item Fractions for methylation of $DACT$ and $WIF1$ genes in the Dickkopf (Jiang et al.) can affect gene expression (Peter et al.)
\end{enumerate}
4. Fractions for methylation and non-methylation status of SFRP1, SFRP2, SFRP4 and SFRP5 (CpG islands around the first exons) was recorded from 6 affected individuals each having both primary CRC tissues and normal colon mucosa from Suzuki et al.\textsuperscript{17} via manual counting through visual inspection of intensity levels from methylation specific PCR (MSP) analysis of gene promoter region and later normalized.

5. Methylation of DACT1 (+52 to +375 BGS) and DACT2 (+52 to +375 BGS) in promoter region for Normal, HT29 and RKO cell lines from Jiang et al.\textsuperscript{3} was recorded via counting through visual inspection of open or closed circles indicating methylation status estimated from bisulfite sequencing analysis and later normalized.

6. Concentration of DVL2 decreases with expression of DACT3 and vice versa Jiang et al.\textsuperscript{3}. Due to lack of exact proportions the probability values were assumed

7. Concentration of $\beta$-catenin given concentrations of DVL2 and DACT1 varies and for static model it is tough to assign probability values. High DVL2 concentration or suppression (expression) of DACT1 leads to increase in concentration of $\beta$-catenin (\textsuperscript{3}, Yuan et al.\textsuperscript{18}). Wet lab experimental evaluations might reveal the factual proportions.

8. Similarly, the concentrations of TRC MPLX (Clevers\textsuperscript{4}, Kriegl et al.\textsuperscript{19}) and TCF4 (Verhaegh et al.\textsuperscript{2}) have been assumed based on their known roles in the Wnt pathway. Actual proportions require further wet lab tests.

9. Finally, the probability of Sample being tumorous or normal is a chance level as it contains equal amount of cancerous and normal cases.

Note that all these probabilities have been recorded in table 1 of Sinha\textsuperscript{1} and their values stored in the probabilities.mat file. Addressing the question of why these probabilities have been estimated earlier, it can be seen that the extra/intracellular factors affecting the Wnt pathway in the data set provided by Jiang et al.\textsuperscript{3} contains some genes whose expression is influenced by epigenetic factors mentioned in table 2. Hence it is important to tabulate and store prior probability values for known biological factors that influence the pathway. Also, the probability values of these nodes have been computed earlier due to prior available information. Once estimated or assumed based on biological knowledge, these probabilities needed not be recomputed and are thus stored in proper format at the beginning of the computational experiment.

2.3 Building the bayesian network model

Next comes the topology of the network using prior biological knowledge made available from results of wet lab experiments documented in literature. This is achieved using the function generateInteraction in the file generateInteraction.m. The function takes in the set of unique genes and the type of model and generates a cell of interaction for the Bayesian network as well as a cell of unique set of Nodenames. interaction contains all the prior established biological knowledge that caries causal semantics in the form of arcs between parent and child nodes. It should be noted that even though the model is not complete due to its static nature, it has the ability to encode prior causal relationships and has the potential for further refinement.

\begin{verbatim}
% Building the Bayesian Network model
%================
% Generate directionality between
% parent and child nodes
[interaction, nodeNames] = ...
    generateInteraction(uniqueGenes,...
    model);
\end{verbatim}

The interaction and nodeNames go as input arguments to the function mk_adj_mat, which generates an adjacency matrix for a directed acyclic graph (DAG) stored in dag. Using functions biograph and input arguments dag and nodeNames generates a structure gObj that can be used to view the topology of the network. A crude representation of $M\textsubscript{PBK+EI}$ and $M\textsubscript{NB+MPBK}$ shown in figures 2 and 3 was generated using the function view.

\begin{verbatim}
% Generate dag for the interaction
% between nodeNames
dag = mk_adj_mat(interaction,...
    nodeNames, 0);

% To visualise the graphs or bayesian
% network
gObj = biograph(dag,nodeNames)
gObj = view(gObj);
\end{verbatim}

Once the adjacency matrix is ready, the initialization of the Bayesian Network can be easily done. The total number of nodes is stored in N and the size of the nodes are defined in nodeSizes. In this project each node has a size of two as they contain discrete values representing binary states. Here the function ones defines a row vector with N columns. Thus each node is set to a size of 2. The total number of discrete nodes is defined in discreteNodes. Finally, the Bayesian Network is created using the function mk_bnet from the BNT that takes the following as input arguments (1) dag - the adjacency matrix (2) nodeSizes - defines the size of the nodes.
Fig. 2 Influence diagram of $M_{PBK+EI}$ contains partial prior biological knowledge and epigenetic information in the form of methylation and histone modification. In this model the state of Sample is distinguished from state of TRCMPLX that constitutes the Wnt pathway.

Fig. 3 Influence diagram of $M_{NB+MPBK}$ is a Naive Bayes model that contains minimal prior biological knowledge. In this model the state of TRCMPLX is assumed to indicate whether the sample is cancerous or not.

and (3) discreteNodes - the vector of nodes with their indices marked to be discrete in the Bayesian Network and dumps the network in the variable bnet.

% BN initialization
N = length(nodeNames); % # of nodes

% Define node sizes. NOTE - nodes are assumed to contain discrete values
nodeSizes = 2*ones(1, N);

% Discrete nodes
discreteNodes = 1:N;

% Create BN
bnet = mk_bnet(dag, nodeSizes, ...
'names', nodeNames, 'discrete', ...
discreteNodes);

Section 4 of Sinha\textsuperscript{1} has been reproduced for completeness in tables 3, 4, 5, 6 and 7.

2.4 Hold out experiment

After the framework of the Bayesian Network has been constructed and initialized, the hold out experiment is conducted. The purpose of conducting the experiment is to generate results on different test data while training the Bayesian Network with different sets of training data, a multiple number of time. From Sinha\textsuperscript{1}, the design of the experiment is a simple 2-holdout experiment where one sample from the normal and one sample from the tumorous are paired to form a test dataset. Excluding the pair formed in an iteration of 2-hold out experiment the remaining samples are considered for train-
Table 3 Bayesian Wnt pathway from Sinha

Network With PBK And EI The NB model (Verhaegh et al.2) assumes that the activation (inactivation) of β-catenin based transcription complex is equivalent to the fact that the sample is cancerous (normal). This assumption needs to be tested and in this research work the two newly improvised models based on prior biological knowledge regarding the signaling pathway assume that sample prediction may not always mean that the β-catenin based transcription complex is activated. These assumptions are incorporated by inserting another node of Sample for which gene expression measurements were available. This is separate from the TRCMPLX node that influences a particular set of known genes in the human colorectal cancer. For those genes whose relation with the TRCMPLX is currently not known or biologically affirmed, indirect paths through the Sample node to the TRCMPLX exist, technical aspect of which will be described shortly. Since all gene expressions have been measured from a sample of subjects the expression of genes is conditional on the state of the Sample. Here both tumorous and normal cases are present in equal amounts. The transcription factor TRCMPLX under investigation is known to operate with the help of interaction between β-catenin with TCF4 and LEF1 (Waterman20, Kriegl et al.19). It is also known that the regions in the TSS of MYC (Yochum21), CCND1 (Schmidt-Ott et al.22), CD44 (Kanwar et al.23), SFRP1 (Caldwell et al.24), WIF1 (Reguart et al.25), DKK1 (González-Sancho et al.26) and DKK4 (Pendas-Franco et al.27, Baehs et al.28) contain factors that have affinity to β-catenin based TRCMPLX. Thus expression of these genes are shown to be influenced by TRCMPLX, in figure 2.

Roles of DKK2 (Matsui et al.29) and DKK3 (Zitt et al.30, Veeck and Dahl31) have been observed in colorectal cancer but their transcriptional relation with β-catenin based TRCMPLX is not known. Similarly, SFRP2 is known to be a target of Pax2 transcription factor and yet it affects the β-catenin Wnt signaling pathway (Brophy et al.32). Similarly, SFRP4 (Feng Han et al.33, Huang et al.34) and SFRP5 (Suzuki et al.9) are known to have affect on the Wnt pathway but their role with TRCMPLX is not well studied. SFRP3 is known to have a different structure and function with respect to the remaining SFRPx gene family (Hoang et al.35). Also, the role of DACT2 is found to be conflicting in the Wnt pathway (Kivimäe et al.36). Thus for all these genes whose expression mostly have an extracellular affect on the pathway and information regarding their influence on β-catenin based TRCMPLX node is not available, an indirect connection has been made through the Sample node. This connection will be explained at the end of this section.

Table 4 Network with PBK+EI from Sinha

Bayesian Wnt pathway Three static models have been developed based on particular gene set measured for human colorectal cancer cases (Jiang et al.3). Available epigenetic data for individual gene is also recorded. For sake of simplicity the models are connoted as $M_{PBK+EI}$ (model with Prior Biological Knowledge (PBK) and Epigenetic Information (EI)), $M_{PBK}$ (model with PBK only) and $M_{NB+MPBK}$ (model with Naive Bayes (NB) formulation and Minimal PBK). All models are simple directed acyclic graphs (DAG) with nodes and edges. Figure 2 shows a detailed influence diagram of $M_{PBK+EI}$ between the nodes and the edges. The nodes specify status of genes expression (DKK1, DKK2, DKK3-1, DKK3-2, DKK4, DACT1, DACT2, DACT3, SFRP1, SFRP2, SFRP3, SFRP4, SFRP5, WIF1, MYC, CD44, CCND1 and LEF1), methylation (MeACT1, MeADTA2, MeSFRP1, MeSFRP2, MeSFRP4, MeSFRP5, MeDKK1, MeDKK4 and MeWIF1), histone marks for DACT3 (H3K27me3 and H3K4me3), transcription complex TRCMPLX, samples Sample and factors involved in formation of TRCMPLX like β-catenin, TCF4 and LEF1. Note that there were two recordings of gene expression DKK3 and thus were distinguished by DKK3−1 and DKK3−2. Some causal relations are based on prior biological knowledge and others are based on assumptions, elucidation of which follows in the next section.

Table 4 Network with PBK+EI from Sinha

Fig. 4 Cases for d-connectivity and d-separation. Black (Gray) circles mean evidence is available (not available) regarding a particular node.
Lastly, it is known that concentration of DVL2 (a member of Disheveled family) is inversely regulated by the expression of DACT3 (Jiang et al.\textsuperscript{3}). High DVL2 concentration and suppression of DACT1 leads to increase in stabilization of $\beta$-catenin which is necessary for the Wnt pathway to be active (Jiang et al.\textsuperscript{3}). But in a recent development (Yuan et al.\textsuperscript{18}) it has been found that expression of DACT1 positively regulates $\beta$-catenin. Both scenarios need to be checked via inspection of the estimated probability values for $\beta$-catenin using the test data. Thus there exists direct causal relations between parent nodes DACT1 and DVL2 and child node, $\beta$-catenin. Influence of methylation (yellow hexagonal) nodes to their respective gene (green circular) nodes represent the affect of methylation on genes. Influence of histone modifications in H3K27me3 and H3K4me3 (blue octagonal) nodes to DACT3 gene node represents the affect of histone modification on DACT3. The $\beta$-catenin (blue square) node is influenced by concentration of DVL2 (depending on the expression state of DACT3) and behavior of DACT1. The aforementioned established prior causal biological knowledge is imposed in the BN model with the aim to computationally reveal unknown biological relationships. The influence diagram of this model is shown in figure 2 with nodes on methylation and histone modification. Another model $\mathcal{M}_{PBK}$ (not shown here) was developed excluding the epigenetic information (i.e removal of nodes depicting methylation and histone modification as well as the influence arcs emerging from them) with the aim to check whether inclusion of epigenetic factors increases the cancer prediction accuracy. In order to understand indirect connections further it is imperative to know about d-connectivity/separability. In a BN model this connection is established via the principle of d-connectivity which states that nodes are connected in a path when there exists no node in the path that has more than one incoming influence edge or there exits nodes in path with more than one incoming influence edge which are observed (i.e evidence regarding such nodes is available) (Charniak\textsuperscript{37}). Conversely, via principle of d-separation nodes are separated in a path when there exists nodes in the path that have more than one incoming influence edge or there exists nodes in the path with at most one incoming influence edge which are observed (i.e evidence regarding such nodes is available). Figure 4 represents three different cases of connectivity and separation between nodes $A$ and $C$ when the path between them passes through node $B$. Connectivity or dependency exists between nodes $A$ and $C$ when $\bullet$ evidence is not present regarding node $B$ in the left graphs of I. and II. in figure 4 or $\bullet$ evidence is present regarding node $B$ in the right graph of III. in figure 4.

### Table 5
Network with PBK+EI continued from Sinha\textsuperscript{1}

| Path | Description |
|------|-------------|
| 1    | SFRP3, Sample, SFRP1, TRCMPLX |
| 2    | SFRP3, Sample, DKK1, TRCMPLX |
| 3    | SFRP3, Sample, WIF1, TRCMPLX |
| 4    | SFRP3, Sample, CD44, TRCMPLX |
| 5    | SFRP3, Sample, DKK4, TRCMPLX |
| 6    | SFRP3, Sample, CCND1, TRCMPLX |
| 7    | SFRP3, Sample, MYC, TRCMPLX |
| 8    | SFRP3, Sample, LEF1, TRCMPLX |
| 9    | SFRP3, Sample, DACT3, DVL2, $\beta$-catenin, TRCMPLX |
| 10   | SFRP3, Sample, DACT1, $\beta$-catenin, TRCMPLX |

Knowledge of evidence regarding nodes of SFRP1 (path 1), DKK1 (path 2), WIF1 (path 3), CD44 (path 4), DKK4 (path 5), CCND1 (path 6) and MYC (path 7) makes Sample and TRCMPLX dependent or d-connected. Further, no evidence regarding state of Sample on these paths instigates dependency or connectivity between SFRP3 and TRCMPLX. On the contrary, evidence regarding LEF1, DACT3 and DACT1 makes Sample (and child nodes influenced by Sample) independent or d-separated from TRCMPLX through paths (8) to (10). Due to the dependency in paths (1) to (7) and the given state of SFRP3 (i.e evidence regarding it being active or passive), the BN uses these paths during inference to find how TRCMPLX might behave in normal and tumorous test cases. Thus, exploiting the properties of d-connectivity/separability, imposing a biological structure via simple yet important prior causal knowledge and incorporating epigenetic information, BN help in inferring many of the unknown relation of a certain gene expression and a transcription complex.

### Table 6
Network with PBK+EI continued from Sinha\textsuperscript{1}
Network With Minimal PBK

Lastly, a Naive Bayes model $M_{NB+MPBK}$ with minimal biological knowledge based on Verhaegh et al.\(^2\) model was also developed with an aim to check if the assumed hypothesis that activation state of $TRCMLPX$ is same as sample being cancerous is correct. In this model all gene expressions are assumed to be transcribed via the $\beta$-catenin based $TRCMLPX$ and thus causal arcs exist from $TRCMLPX$ to different gene nodes. The complex itself is influenced by $\beta$-catenin and TCF4 only. Such models can be used for prediction purpose but are not useful in revealing hidden biological relationships as no or minimal prior biological information is imposed on the Naive Bayes model. Figure 3 shows the Naive Bayes model.

Table 7 Network with NB+MPBK from Sinha\(^1\)

| 
| --- |

in the whole data set as well as the search of the indemnies of the labels. For this the values in the variable $\text{noSamples}$ and $\text{transGroundTruthLabels}$ computed from function $\text{readCustomFile}$ is used. $\text{noPos}$ ($\text{noNeg}$) and $\text{posLabelIdx}$ ($\text{negLabelIdx}$) store the number of positive (negative) labels and their indemnies, respectively.

```matlab
% Hold out expriment
%======
% Compute no. of positive and negative % labels and find indemnies of both
noPos = 0;
posLabelIdx = [];
negLabelIdx = [];
for i = 1:noSamples
    if transGroundTruthLabels(i) > 0
        noPos = noPos + 1;
posLabelIdx = [posLabelIdx, i];
    else
        noNeg = noNeg + 1;
egLabelIdx = [negLabelIdx, i];
    end
end

For storing results as well as the number of times the experiment will run, variables $\text{runCnt}$ and $\text{Runs}$ are initialized. The condition in the if statement is not useful now and will be described later.

runCnt = 0;
Runs = struct([]);
if isempty(strfind(evidence, 'me'))
    RunsOnObservedMethylation = ...
    struct([]);
end

for each and every positive (cancerous) and negative (normal) labels, the number of times the experiments runs is incremented in the count variable $\text{runCnt}$. Next the indemnies for test data is separated by using the $i^{th}$ positive and the $j^{th}$ negative label and its indemnies is stored in $\text{testDataIdx}$. The test data itself is then separated from $\text{expressionMatrix}$ using $\text{testDataIdx}$ and stored in $\text{dataForTesting}$. The corresponding ground truth labels of the test data are extracted from $\text{transGroundTruthLabels}$ using $\text{testDataIdx}$ and stored in $\text{labelForTesting}$.

for i = 1:noPos
    for j = 1:noNeg
        % Count for number of runs
        runCnt = runCnt + 1;
        % Build test dataset (only 2
% examples per test set
 testDataIdx = [negLabelIdx(j),...  
                 posLabelIdx(i)];
 dataForTesting = expressionMatrix(:,...
                 testDataIdx);
 labelForTesting = ...  
 transGroundTruthLabels(:,...
                 testDataIdx);

After the storage of the test data and its respective indices, trainingDataIdx is used to store the indices of training data by eliminating the indices of the test data. This is done using temporary variables tmpPosLabelIdx and tmpNegLabelIdx. trainingDataIdx is used to store the training data in variable dataForTraining using expressionMatrix and the indices of training data in variable labelForTraining using transGroundTruthLabels.

% Remove test dataset from the whole % dataset and build train dataset tmpPosLabelIdx = posLabelIdx;
 tmpNegLabelIdx = negLabelIdx;
 tmpPosLabelIdx(i) = [];
 tmpNegLabelIdx(j) = [];
 trainDataIdx = [tmpNegLabelIdx,...  
                 tmpPosLabelIdx];
 dataForTraining = ...  
 expressionMatrix(:,trainDataIdx);
 labelForTraining = ...  
 transGroundTruthLabels(:,...
                 trainDataIdx);

2.4.1 Defining and estimating probabilities and conditional probabilities tables for nodes in bnet: Till now, the probabilities as well as conditional probability tables (cpt) for some of the nodes have been stored in the probabilities.mat file and loaded in the workspace. But the cpt for all the nodes in the bnet remain uninitialized. The next procedure is to initialize the tables using assumed values for some of the known nodes while estimating the entries of cpt for other nodes using training data.

To this end it is important to define a variable by the name cpdStorage of the format structure. Starting with all the nodes that have no parents and whose probabilities and cpt have been loaded in the workspace (saved in probabilities.mat), the for loop iterates through all the nodes in the network defined by N, stores the index of the $k^{th}$ node in nodeidx using function bnet.names with input argument nodeNames{k} and assigns values to cpt depending on the type of model. If $M_{PBK+EI}$ (model='t2') is used and the $k^{th}$ entry in nodeNames matches with TCF4 then the cpt value in PrTCF4 is assigned to cpt. The parent node of this node is assigned a value 0 and stored in cpdStorage(k).parentnode{i}. The name TCF4 or nodeNames{k} is assigned to cpdStorage(k).node. The cpt values in cpt is assigned to cpdStorage(k).cpt. Finally, the conditional probability density cpt for the node with name TCF4 is stored in bnet.CPD using function tabular_CPD, the Bayesian Network bnet, the node index nodeidx and cpt. Similarly, values in PrMeDKK1, avgPrMeDACT1, avgPrMeDACT2, avgPrH3K27me3, avgPrH3K4me3, PrMeSFRP1, PrMeSFRP2, PrMeSFRP4, PrMeSFRP5, PrMeWIF1 and PrSample initialize the cpt values for nodes MeDACT1, MeDACT2, H3k27me3, H3k4me3, MeSFRP1, MeSFRP2, MeSFRP4, MeSFRP5, MeWIF1 and Sample, respectively.

Similar initializations happen for models $M_{PBK}$ (model='t2') and $M_{NB+MPBK}$ (model='p1'). It should be noted that in $M_{PBK}$ ($M_{NB+MPBK}$) the only nodes without parents are TCF4 and Sample (TCF4 and BETACAT). To accommodate for these models, the necessary elseif statements have been embedded in the for loop below.

% Define P and CPD for the nodes of the % bnet cpdStorage = struct([]);
 % Store probabilities for nodes with no % parents for k = 1:N
 nodeidx = bnet.names(nodeNames{k});
 if isempty(bnet.parents(nodeidx))
 % tables for non-gene measurements
     if ~isempty(strfind(model,'t1'))
         if strcmp(nodeNames{k},’TCF4’)
             cpt = PrTCF4;
         elseif strcmp(nodeNames{k},’MeDACT1’)
             cpt = PrMeDACT1;
         elseif strcmp(nodeNames{k},’MeDACT2’)
             cpt = avgPrMeDACT2;
         elseif strcmp(nodeNames{k},’H3k27me3’)
             cpt = avgPrH3K27me3;
         elseif strcmp(nodeNames{k},’MeSFRP1’)
             cpt = PrMeSFRP1;
         else
             cpt = PrMeSFRP1;
     else
         if strcmp(nodeNames{k},’TCF4’)
             cpt = PrTCF4;
         elseif strcmp(nodeNames{k},’MeDACT1’)
             cpt = PrMeDACT1;
         elseif strcmp(nodeNames{k},’MeDACT2’)
             cpt = avgPrMeDACT2;
         elseif strcmp(nodeNames{k},’H3k4me3’)
             cpt = avgPrH3K4me3;
         elseif strcmp(nodeNames{k},’MeSFRP1’)
             cpt = PrMeSFRP1;
         else
             cpt = PrMeSFRP1;
     end
end
elseif strcmp(nodeNames{k},... 'MeSFRP2')
cpt = PrMeSFRP2;
elseif strcmp(nodeNames{k},... 'MeSFRP4')
cpt = PrMeSFRP4;
elseif strcmp(nodeNames{k},... 'MeSFRP5')
cpt = PrMeSFRP5;
elseif strcmp(nodeNames{k},... 'MeWIF1')
cpt = PrMeWIF1;
elseif strcmp(nodeNames{k},... 'Sample')
cpt = PrSample;
end
elseif ~isempty(strfind(model,... 't2'))
if strcmp(nodeNames{k},'TCF4')
cpt = PrTCF4;
elseif strcmp(nodeNames{k},... 'Sample')
cpt = PrSample;
end
elseif ~isempty(strfind(model,... 'p1'))
if strcmp(nodeNames{k},'TCF4')
cpt = PrTCF4;
elseif strcmp(nodeNames{k},... 'BETACAT')
cpt = PrBETACAT;
end
end
cpdStorage(k).parentnode{1} = 0;
cpdStorage(k).node = nodeNames(k);
cpdStorage(k).cpt = cpt;
bnet.CPD(nodeidx) = tabular_CPD(... bnet, nodeidx, 'CPT', cpt);
end
end

In the same for loop above, the next step is to initialize probability as well as the cpt values for nodes with parents. Two cases exist in the current scenario, i.e nodes that (1) represent genes and (2) do not represent genes. To accommodate for gene/non-gene node classification a logical variable GENE is introduced. Also, before entering the second for loop described below, a variable gene_cpd of the format structure is defined for storage of the to be computed cpt values for all genes in the data set. parentidx stores the index of the parents of the child node under consideration using the child’s index in nodeidx via bnet.parents{nodeidx}.

The total number of parents a child node has is contained in noParents.

Initially GENE is assigned a value of 0 indicating that the node under consideration is not a gene node. If this is the case, the `GENE in the if condition of the for loop below gets executed. In this case, depending on the type of the model cpt values of a particular node is initialized. For \( M_{PBK+E1} \) and \( M_{PBK} \) (model='t1' and model='t2'), the cpt values for nodes BETACAT, DVL2 and TRCMPLX is stored using values in PrBETACAT, PrDVL2 and PrTRCMPLX. As before, using function tabular_CPD and values in nodeidx, bnet and cpt as input arguments, the respective cpt is initialized in bnet.CPD{nodeidx}. Similar computations are done for \( M_{NB+PBK} \) i.e model 'p1' for node TRCMPLX. Finally, the indices of the parents of the \( k^{th} \) child node is stored in cpdStorage(k).parentnode{m].

On the other hand, if the name of the node in the \( k^{th} \) index of nodeNames matches the name in the \( l^{th} \) index of uniqueGenes, a parent variable of format cell is defined within the second nested for loop below. The names of the parents are stored in this variable using nodeNames{parentidx(n)}. Next, the cpt values of these parent nodes are separately stored using a cell parent_cpd and a count cnt. Finally, the cpt values for the \( l^{th} \) gene is determined using the function generateGeneCpd in the script generateGeneCpd.m that takes the following input arguments (1) vecTraining - gene expression of from training data (2) labelTraining - labels for training data (3) nodeName - name of the gene involved (4) parent - name of parents of the child node or the gene under consideration (5) parent_cpd - parent cpd values (6) model - kind of model and finally returns the output as a structure gene_cpd containing cpd for the particular gene under consideration given its parents as well as a threshold value in the form of median. In the code below, the values of the following variables go as input arguments for the function generateGeneCpd, in order (1) dataForTraining(l,:) - training data for the \( l^{th} \) unique gene, (2) labelForTraining - labels for training data, (3) uniqueGenes{l}, (4) parent, (5) parent_cpd, (6) model. The output of the function is stored in the structure variable x. The threshold at which the probabilities were computed for the \( l^{th} \) gene is stored in gene_cpd(l).vecmedian using x.vecmedian and the probabilities themselves are stored in gene_cpd(l).T using x.T. These probabilities are reshaped into a row vector and stored in cpt. As mentioned before, using function tabular_CPD and values in nodeidx, bnet and cpt as input arguments, the respective cpt is initialized in bnet.CPD{nodeidx}. Finally, required values of cpt, name of \( l^{th} \) gene or \( k^{th} \) node and indices of its parent nodes are stored in cpdStorage(k).cpt, cpdStorage(k).node and
It should be noted that the exposition of the generation of probability values for the different genes via the function generateGenecpd needs a separate treatment and will be addressed later. To maintain the continuity of the workflow of the program, the next step is addressed after the code below.

```matlab
% Store probabilities for nodes with % parents
gene_cpd = struct([]);
for k = 1:N
    nodeidx = bnet.names(nodeNames{k});
    if ~isempty(bnet.parents{nodeidx})
        parentidx = bnet.parents{nodeidx};
        noParents = length(parentidx);
        GENE = 0;
        for l = 1:noGenes
            if strcmp(nodeNames{k},...
                uniqueGenes{l})
                % Find cpt of gene parent
                parent = {();
                for n = 1:noParents
                    parent{n} = ...
                        nodeNames{parentidx(n)};
                end
                % Assign cpd to parent
                cnt = 0;
                parent_cpd = {();
                for m = 1:length(cpdStorage)
                    for n = 1:noParents
                        if strcmp(parent{n},...
                            cpdStorage(m).node)
                            cnt = cnt + 1;
                            parent_cpd{cnt} = ...
                                cpdStorage(m).cpt;
                    end
                end
                x = generateGenecpd(...
                    dataForTraining(l,:),...,%
                    labelForTraining,...
                    uniqueGenes{l}, parent,...
                    parent_cpd, model);
                gene_cpd(l).vecmedian = ...
                    x.vecmedian;
                gene_cpd(l).T = x.T;
                [r, c] = size(gene_cpd(l).T);
                cpt = reshape(gene_cpd(l).T,1,r*c);
                GENE = 1;
                break;
            end
        end
        if ~GENE
            if ~isempty(strfind(model,'t1'))
                if strcmp(nodeNames{k},'BETACAT')
                    cpt = PrBETACAT;
                elseif strcmp(nodeNames{k},'DVL2')
                    cpt = PrDVL2;
                elseif strcmp(nodeNames{k},'TRCMPLX')
                    cpt = PrTRCMPLX;
                end
            elseif ~isempty(strfind(model,'t2'))
                if strcmp(nodeNames{k},'BETACAT')
                    cpt = PrBETACAT;
                elseif strcmp(nodeNames{k},'DVL2')
                    cpt = PrDVL2;
                elseif strcmp(nodeNames{k},'TRCMPLX')
                    cpt = PrTRCMPLX;
                end
            elseif ~isempty(strfind(model,'p1'))
                if strcmp(nodeNames{k},'TRCMPLX')
                    cpt = PrTRCMPLX;
                end
            end
        end
        for m = 1:noParents
            cpdStorage(k).parentnode{m} = ...
                parentidx(m);
        end
        cpdStorage(k).node = nodeNames{k};
        cpdStorage(k).cpt = cpt;
        bnet.CPD{nodeidx} = ...
            tabular_CPD(bnet,nodeidx,'CPT',cpt);
    end
end
end
end
```

### 2.4.2 Evidence building and inference

The values estimated in `gene_cpd` as well as `cpdStorage` are stored for each and every run of the hold out experiment. Also, the dimensions of the testing data is stored.

```matlab
% Function to store estimated % parameters
Runs(runCnt).geneCpd = gene_cpd;
Runs(runCnt).cpdStorage = cpdStorage;

% Function to predict on test data % using trained BN
[r, c] = size(dataForTesting);
Next, depending on the type of the evidence provided in eviDence, inferences can be made. Below, a section of code for evidence gene expression, which gets executed when
the case 'ge' matches with the parameter evidence of the switch command, is explained. The issue that was to be investigated was whether the β-catenin based TRCMPLX is always switched on (off) or not when the Sample is cancerous (normal). In order to analyze this biological issue from a computational perspective, it would be necessary to observe the behaviour of the predicted states of both TRCMPLX as well as Sample, given all the available evidence. For this purpose, variable tempTRCMPLXgivenAllge is defined as a vector for each model separately, while variable tempSAMPLE is defined as a vector for biologically inspired models i.e $M_{PBK+EI}$ and $M_{PBK}$ separately. This is so due to the assumption that the state of TRCMPLX is the same as the state of the test sample under consideration in the $M_{NB+MPBk}$ (a modification of Verhaegh et al.\(^2\)).

In the section of the code below, for each of the test dataset an evidence variable of the format cell is defined. The evidence is of the size of equivalent to the number of nodes in the network. Only those indices in the cell will be filled for which information is available from the test data. Since the function twoHoldOutExp started with 'ge' as an argument for type of evidence, evidence will be constructed from information available via gene expression from the test data. Thus for the \(m^\text{th}\) gene, if the gene expression in the test data (i.e dataForTesting\((m,k)\)) is lower than the threshold generated using the median of expressions for this gene in the training data (i.e gene\_cpd\((m)\).vecmedian), then the evidence for this gene is considered as inactive or repressed, i.e evidence\{bnet.names\(\{\text{uniqueGenes}(m)\}\)\} = 1, else the evidence for this gene is considered active or expressed i.e evidence\{bnet.names\(\{\text{uniqueGenes}(m)\}\)\} = 2. Iterating through all the genes, the evidence is initialized with the available information for the \(k^\text{th}\) test data.

Once the probability values have been initialized either by computation or assumption, then for the \(k^\text{th}\) test data, a Bayesian network engine is generated and stored in bnetEngine via the junction tree algorithm implemented in function jtree\_inf\_engine that uses the input argument as the newly initialized network stored in bnet. The bnetEngine is then fed with the values in evidence to generate a new engine that contains the updated probability values for nodes without evidence in the network. This is done using the function enter\_evidence. According to BNT provided by Murphy et al.\(^5\), in the case of the jtree engine, enter\_evidence implements a two-pass message-passing scheme. The first return argument (engine) contains the modified engine, which incorporates the evidence. The second return argument (loglik) contains the log-likelihood of the evidence. It is the first returned argument or the modified engine that will be of use further. It is important to note that for every iteration that points to a new test data in the for loop, a new Bayesian network engine is generated and stored in bnetEngine. If this is not done, then the phenomena of explaining away can occur on feeding new evidence to an already modified engine which incorporated the evidence from the previous test data. In explaining away the entrence of new evidence might out weigh the effect of an existing influencing factor or evidence thus making the old evidence redundant. This simulation is not related to such study of explaining away.

Finally, the belief that the TRCMPLX is switched on given the gene expression evidence i.e $Pr(TRCMPLX = 2|\text{ge as evidence})$ is computed by estimating the marginal probability values using the function marginal\_nodes which takes the engine stored in engine and the name of the node using bnet\_names(‘TRCMPLX’). The marginal probabilities are stored in margTRCMPLX. The final probability of $TRCMPLX$ being switched on given all gene expression evidences is stored in tempTRCMPLXgivenAllge using margTRCMPLX.T(2). Similarly, for biologically inspired models the belief that the test Sample is cancerous given the gene expression evidence i.e $Pr(\text{Sample} = 2|\text{ge as evidence})$ is computed using function marginal\_nodes that takes the engine stored in engine and the name of the node using bnet\_names(‘Sample’). The marginal probabilities are stored in margSAMPLE. The final probability of Sample being cancerous given all gene expression evidences is stored in tempSAMPLE using margSAMPLE.T(2).

```matlab
switch evidence
  case 'ge'
    disp(['Testing Example ',...
         num2str(runCnt), ' - Based on all ge']);
    tempTRCMPLXgivenAllge = [];
    if isempty(strfind(model, 't'))
      tempSAMPLE = [];
    end
    % Build evidence for inference
    for k = 1:c
      evidence = cell(1,N);
      for m = 1:noGenes
        if dataForTesting\((m,k)\) <= ...
          gene\_cpd(m).vecmedian
          evidence\{bnet\_names(...
            uniqueGenes(m))\} = 1;
        else
          evidence\{bnet\_names...
            (uniqueGenes(m))\} = 2;
        end
      end
      % Build Bayesian engine
      bnetEngine = jtree\_inf\_engine(bnet);
      [engine, loglik] = ...
```
condPrSAMPLE and condPrTRCMPLX given all gene expression evidence, a cell variable Results, a counter cntResult and vector variables condPrTRCMPLXgivenAllge, condPrSAMPLE and labels are defined as well as initialized. Next, the prediction values and original labels are stored while iterating through the total number of runs of the experiment. This is done using the for loop and the variable runCnt. For the $i^{th}$ run, predicted conditional probabilities of TRCMPLX and Sample from each run is stored in condPrTRCMPLXgivenAllge($i,:$) and condPrSAMPLE($i,:$), depending on the model used. Finally, the ground truth labels of the test data are stored in a matrix were the $i^{th}$ row is initialized with labels($i,:$) = [-1, +1]; Here, labels it a matrix and -1 (+1) represent normal (cancerous) cases. Next, the variables condPrTRCMPLXgivenAllge and condPrSAMPLE are reshaped into vectors for further processing.

The plotting of the ROC curves and the estimation of their respective AUCs is achieved using function perfcurve that takes labels, either of the vectors condPrTRCMPLXgivenAllge or condPrSAMPLE depending on the type of model selected. The function churns out useful information in the form of the false positive rate in $X$, the true positive rate in $Y$ and the estimated AUC for ROC of condPrTRCMPLXgivenAllge (condPrSAMPLE) in AUCTRCMPLXgivenAllge (AUCSAMPLE). The plot function is used to draw the graphs along with the depiction of legends using function legend. Finally, the two sample Kolmogorov-Smirnov test between the predictions of states of TRCMPLX and Sample is performed using the kstest2 function. This function takes the two vectors condPrTRCMPLXgivenAllge and condPrSAMPLE as arguments, compares the distribution of the predictions and returns the state of significance between the two distributions in h01. If the value of h01 is 1, then statistical significance exists else it does not exist. Sinha shows that the statistical difference exists between predictions of TRCMPLX and Sample when the nodes for the same are segregated in the biologically inspired causal models, which is not the case with the naive Bayes model.

Lastly, the computed variables are stored in a .mat file using the function save. Options for using the save function can be obtained from the help command in Matlab.

2.5 Storing results, plotting graphs and saving files

The final section of the code deals with storing of the results, plotting of graphs and saving the results in the files. Since the current explanation is for gene expression evidence, the code pertaining to ‘ge’ is explained. Readers might want to develop the code for evidence regarding methylation as a programming project.

To store results as well as the conditional probabilities for TRCMPLX and SAMPLE given all the gene expression evidence, a cell variable Results, a counter cntResult and vector variables condPrTRCMPLXgivenAllge, condPrSAMPLE and labels are defined as well as initialized.
% original labels
for i = 1:runCnt
    condPrTRCMPLXgivenAllge(i,:) = Runs(i).condPrTRCMPLXgivenAllge;
geneEvidence(i) = Runs(i).geneEvidence;
    if ~isempty(strfind(model,'t'))
        condPrSAMPLE(i,:) = Runs(i).condPrSAMPLE;
    end
    labels(i,:) = [-1, +1];
end

% Reshape the vectors
[r,c] = size(labels);
labels = reshape(labels,r*c,1);
condPrTRCMPLXgivenAllge = reshape(condPrTRCMPLXgivenAllge,...
    r*c,1);
if ~isempty(strfind(model,'t'))
    condPrSAMPLE =...
        reshape(condPrSAMPLE,r*c,1);
end
% Plot the ROC curve and compute AUC
[X,Y,T,AUCTRCMPLXgivenAllge] =...  
    perfcurve(labels,...
        condPrTRCMPLXgivenAllge,1);
plot(X,Y,'r');
xlabel('False positive rate');
ylabel('True positive rate');
    if ~isempty(strfind(model,'t'))
        hold on;
        [X,Y,T,AUCSAMPLE] =...
            perfcurve(labels,condPrSAMPLE,1);
        plot(X,Y,'b');
        legend('TRCMPLX - On','SAMPLE - T');
        hold off;
    end

% Perform ks-test the significance
% between models/evidences/predictions
[h01,p,ksstat] =...
    ktest2(condPrTRCMPLXgivenAllge,...
        condPrSAMPLE);
end

if ~isempty(strfind(model,'t1'))
    save('Results.mat','Runs',...  
        'condPrTRCMPLXgivenAllge',...
        'geneEvidence',...  
        'condPrSAMPLE',...
        'AUCTRCMPLXgivenAllge',...  
        'AUCSAMPLE',...  
        'h01');
elseif ~isempty(strfind(model, 'p1'))
    save('Results.mat','Runs',...  
        'condPrTRCMPLXgivenAllge',...
        'geneEvidence',...
        'AUCTRCMPLXgivenAllge');
end
else
end

The ROC graphs and their respective AUC values found in the figures of Sinha\textsuperscript{1} are plotted by making variation in the assumed probability values of PrTRCMPLX in the function generateGenecpd that will be discussed later. Interpretations of the results can be studied in more depth from Sinha\textsuperscript{1}.

Finally, a full section is dedicated to the computation of the probabilities for nodes with parents which has been implemented in function generateGenecpd.

2.6 Generating probabilities for gene nodes with parents

Here, the code for the function generateGenecpd is explained. As a recapitulation, the function generateGenecpd in the script generateGenecpd.m takes the following input arguments (1) vecTraining - gene expression of from training data (2) labelTraining - labels for training data (3) nodeName - name of the gene involved (4) parent - name of parents of the child node or the gene under consideration (5) parent_cpd - parent cpd values (6) model - kind of model and finally returns the output as a structure gene_cpd containing cpd for the particular gene under consideration given its parents as well as a threshold value in the form of median. In the code below, the values of the following variables go as input arguments for the function generateGenecpd, in order (1) dataForTraining(l,:)- training data for the l\textsuperscript{th} unique gene, (2) labelForTraining - labels for training data (3) uniqueGenes{l}, (4) parent, (5) parent_cpd, (6) model. The output of the function is stored in the structure variable x. The threshold at which the probabilities were computed for the l\textsuperscript{th} gene is stored in gene_cpd(l).vecmedian using x.vecmedian and the probabilities themselves are stored in gene_cpd(l).T using x.T.

The code begins with the storing of the dimension of a gene expression vector in vecTraining in r and c and recording the length of the vector containing the labels for the training data (in labelTraining) in lencond. Finally, the much reported threshold is estimated here using the median of the training data and stored in vecmedian.
the cpt values for the table is segregated based on the status of different parent factors affecting the gene under investigation. This is also used along with the labels of the training data to decide the frequency of occurrence of the gene in different conditions. The median of the expression values is employed as a threshold to decide the frequency of occurrence where the gene is inactive. In case of expression where the gene is active and inactive is represented by vector mINn and (d) un-methylated in tumorous (represented by vector umINt), cases. For every jth entry in the vecTraining, if the label (labelTraining(j)) is normal (≤0) and the DKK1 gene expression (vecTraining(j)) is less than the estimated median (≥vecmedian) then the value in vecTraining(j) is appended to mINn. Here, expression level lower than median indicates probable repression due to methylation in normal case. If the label (labelTraining(j)) is tumorous (≥0) and the DKK1 gene expression (vecTraining(j)) is greater than the estimated median (≤vecmedian) then the value in vecTraining(j) is appended to umINt. Here, expression level greater than median indicates probable activation due to un-methylation in normal case. If the label (labelTraining(j)) is tumorous (≥0) and the DKK1 gene expression (vecTraining(j)) is less than the estimated median (≥vecmedian) then the value in vecTraining(j) is appended to mINn. Here, expression level lower than median indicates probable repression due to methylation in tumorous case. And finally, if the label (labelTraining(j)) is tumorous (≥0) and the DKK1 gene expression (vecTraining(j)) is greater than the estimated median (≥vecmedian) then the value in vecTraining(j) is appended to umINt. Here, expression level greater than median indicates probable activation due to un-methylation in tumorous case.

% Segregate values based on status of methylation and samples

mINn = []; umINn = []; mINt = []; umINt = [];
for j = 1:lencond
    if labelTraining(j) < 0 && ... vecTraining(j) < vecmedian
        mINn = [mINn, vecTraining(j)];
    elseif labelTraining(j) < 0 && ... vecTraining(j) >= vecmedian
        umINn = [umINn, vecTraining(j)];
    elseif labelTraining(j) > 0 && ... vecTraining(j) < vecmedian
        mINt = [mINt, vecTraining(j)];
    else
        umINt = [umINt, vecTraining(j)];
end

% Defining affect of TRCMPLX on % gene expression

noParents = length(parent);
for i = 1:noParents
    if isempty(strfind(model,'t'))
        if strfind(parent(i),'TRCMPLX')
            p = 0.5;
        end
    end
end

In Sinha et al., the effect of TRCMPLX on the gene expression has been analysed as it is not known to what degree the TRCMPLX plays a role in the Wnt signaling pathway. To investigate this Sinha et al. incorporated a parameter p that encodes the effect of TRCMPLX on the expression of the gene which is influenced by it. Thus while iterating through the list of parents if one encounters TRCMPLX as a parent, then p is initialized to a certain value. In Sinha et al., the effect of TRCMPLX being active (1 − p) is incremented in steps of 0.1 from {0.5 to 0.9} and respective ROC graphs are plotted using the same.

2.6.1 **DKK1:** (model='t1') Since there are three parents for DKK1, namely MeDKK1, Sample and TRCMPLX, the cpt values for the table is segregated based on the status of methylation and quality of samples. A 2 × 2 cross table for methylation and sample generates frequency estimates that can help derive probability values. The entries of the cross table depict the following cases (a) methylated in normal (represented by vector mINn) (b) un-methylated in normal (represented by vector umINn) (c) methylated in tumorous (represented by vector mINt) and (d) un-methylated in tumorous (represented by vector umINt), cases. For every jth entry in the vecTraining, if the label (labelTraining(j)) is normal (≤0) and the DKK1 gene expression (vecTraining(j)) is less than the estimated median (≥vecmedian) then the value in vecTraining(j) is appended to mINn. Here, expression level lower than median indicates probable repression due to methylation in normal case. If the label (labelTraining(j)) is tumorous (≥0) and the DKK1 gene expression (vecTraining(j)) is greater than the estimated median (≤vecmedian) then the value in vecTraining(j) is appended to umINt. Here, expression level greater than median indicates probable activation due to un-methylation in normal case. If the label (labelTraining(j)) is tumorous (≥0) and the DKK1 gene expression (vecTraining(j)) is less than the estimated median (≤vecmedian) then the value in vecTraining(j) is appended to mINn. Here, expression level lower than median indicates probable repression due to methylation in tumorous case. And finally, if the label (labelTraining(j)) is tumorous (≥0) and the DKK1 gene expression (vecTraining(j)) is greater than the estimated median (≥vecmedian) then the value in vecTraining(j) is appended to umINt. Here, expression level greater than median indicates probable activation due to un-methylation in tumorous case.

% Rows is the gene expression and... % columns are conditions (normal or cancerous)
[r, c] = size(vecTraining);
lencond = length(labelTraining);

% Take median as the threshold
vecmedian = median(vecTraining);

% Defining affect of TRCMPLX on % gene expression
noParents = length(parent);
for i = 1:noParents
    if isempty(strfind(model,'t'))
        if strfind(parent(i),'TRCMPLX')
            p = 0.5;
        end
    end
end
\begin{align*}
\text{Before estimating the values for } & \text{cpt of } DKK1, \text{ it is important to see how (1) the probability table would look like and (2) the probability table is stored in BNT (Murphy et al.\textsuperscript{15}). Table 8 represents the conditions of sample as well as the methylation along with transcription complex and the probable beliefs of events } (DKK1 \text{ being on/off). With three parents and binary state, the total number of conditions is } 2^3. \text{ To estimate the values of the probable beliefs of an event, the following computation is done. (Case } - \text{ TRCMPLX is Off) The Pr(DKK1 - Off|Sample - Normal,Me - UM) being low, is the fraction of number of 1's in the normal sample } (a \times p) \text{ and the sum of total number of normal samples and number of 1's in the tumorous samples, i.e the non-methylated gene expression values in tumorous samples } (b). \text{ Similarly, Pr(DKK1 - On|Sample - Tumor,Me - UM) being low, is the fraction of number of 1's in the tumorous sample } (b \times p) \text{ and the sum of total number of tumorous samples and number of 1's in the normal samples, i.e the non-methylated gene expression values in normal samples } (B). \text{ Again, Pr(DKK1 - Off|Sample - Normal,Me - M) being high, is the fraction of number of 0's in the normal sample } (c \times p) \text{ and the sum of total number of normal samples and number of 0's in the tumorous samples, i.e the methylated gene expression values in tumorous samples } (C). \text{ Finally, Pr(DKK1 - Off|Sample - Tumor,Me - M) being high, is the fraction of number of 0's in the tumorous sample } (d \times p) \text{ and the sum of total number of tumorous samples and number of 0's in the normal samples, i.e the methylated gene expression values in normal samples } (D). \\text{ (Case } - \text{ TRCMPLX is On) Next, the Pr(DKK1 - On|Sample - Normal,Me - UM) being low, is the fraction of number of 1's in the normal sample } (a \times (1 - p)) \text{ and the sum of total number of normal samples and number of 1's in the tumorous samples, i.e the non-methylated gene expression values in tumorous samples } (A). \text{ Similarly, Pr(DKK1 - On|Sample - Tumor,Me - UM) being low, is the fraction of number of 1's in the tumorous sample } (b \times (1 - p)) \text{ and the sum of total number of tumorous samples and number of 1's in the normal samples, i.e the non-methylated gene expression values in normal samples } (B). \text{ Again, Pr(DKK1 - Off|Sample - Normal,Me - M) being high, is the fraction of number of 0's in the normal sample } (c \times (1 - p)) \text{ and the sum of total number of normal samples and number of 0's in the tumorous samples, i.e the methylated gene expression values in tumorous samples } (C). \text{ Finally, Pr(DKK1 - Off|Sample - Tumor,Me - M) being high, is the fraction of number of 0's in the tumorous sample } (d \times (1 - p)) \text{ and the sum of total number of tumorous samples and number of 0's in the normal samples, i.e the methylated gene expression values in normal samples } (D). \text{ Complementary conditional probability values for } DKK1 \text{ being inactive can easily be computed from the above estimated values.}
\end{align*}

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|}
\hline
Sample & Methylation & TRCMPLX & Pr(DKK1=Off) & Pr(DKK1=On) \\
\hline
\hline
Normal & No & Off & h (1) & l (9) \\
\hline
Normal & Yes & Off & h (3) & l (11) \\
\hline
Normal & No & On & h (5) & l (13) \\
\hline
Tumor & No & On & h (3) & l (13) \\
\hline
Tumor & Yes & On & h (7) & l (15) \\
\hline
Tumor & Yes & Off & h (8) & l (16) \\
\hline
\end{tabular}
\caption{Conditional probability table for \textit{DKK1} in \textit{M_{PBK+EI}} (model-t1). h - probability of event being high; l - probability of event being low. Serial numbers in brackets represent the ordering of numbers in vectorial format.}
\end{table}
ues in T has been established, a constant 1 is added as pseudo count to convert the distribution to a probability distribution via Dirichlet process. This is done to remove any deterministic 0/1 values appearing in the probability tables. If 0/1 appears in the probability tables then one has deterministic evidence regarding an event and the building of the Bayesian engine collapses. Finally, the frequencies in T are normalized in order to obtain the final conditional probability values for DKK1. Estimation of cpts for genes SFRP1, WIF1 and DKK4 which have methylation, TRCMPLX and Sample as parents require same computations as above. Figure 5 shows the pictorial representation of one of the cpt in \( M_{PBK+EI} \).

% Multiply probability of TRCMPLX in % on/off state to add the 3rd % dimension in deciding the conditional % probability tables.

% Conditional probability table for % DKK1 given its parents
T = [A-a*p, a*p;...
B-b*p, b*p;...
c*p, C-c*p;...
d*p, D-d*p;...
A-a*(1-p), a*(1-p);...
B-b*(1-p), b*(1-p);...
c*(1-p), C-c*(1-p);...
d*(1-p), D-d*(1-p)];
[r,c] = size(T);

% Convert the table to probability % distribution via Dirichlet process
T = T + 1;
for i = 1:r
T(i,:) = T(i,:)./sum(T(i,:));
end
(model='t2') There are two parents for DKK1, namely TRCMPLX and Sample. The conditional probability value for a gene being active or inactive is estimated based on the state of the Sample. But since the actual probability values for the activation of the TRCMPLX is not known the conditional probabilities are multiplied with a probability value of \( p \) when the TRCMPLX is off and with probability value \( 1-p \) when the TRCMPLX is on.

The analysis of quality of sample generates frequency estimates that can help derive probability values. These frequencies depict the following cases (a) gene repressed in normal (represented by vector \( \text{offINn} \)) (b) gene expressed in normal (represented by vector \( \text{onINn} \)) (c) gene repressed in tumorous (represented by vector \( \text{offINt} \)) and (d) gene expressed in tumorous (represented by vector \( \text{onINt} \)), cases. For every \( j \)th entry in the vecTraining, if the label (\( \text{labelTraining(j)} \)) is normal (\( \leq 0 \)) and the DKK1 gene expression (\( \text{vecTraining(j)} \)) is less than the estimated median (\( \leq \text{vecmedian} \)) then value in \( \text{vecTraining(j)} \) is appended to \( \text{offINn} \). Here, expression level lower than median indicates probable gene repression in normal case. If the label (\( \text{labelTraining(j)} \)) is normal (\( \leq 0 \) and the
**DKK1** gene expression ($\text{vecTraining}(j)$) is greater than the estimated median ($\geq \text{vecmedian}$) then value in $\text{vecTraining}(j)$ is appended to $\text{onINn}$. Here, expression level greater than median indicates probable gene activation in normal case. If the label ($\text{labelTraining}(j)$) is tumorous ($\geq 0$) and the **DKK1** gene expression ($\text{vecTraining}(j)$) is less than the estimated median ($\leq \text{vecmedian}$) then value in $\text{vecTraining}(j)$ is appended to $\text{offINt}$. Here, expression level lower than median indicates probable gene repression in tumour case. And finally, If the label ($\text{labelTraining}(j)$) is tumorous ($\geq 0$) and the **DKK1** gene expression ($\text{vecTraining}(j)$) is greater than the estimated median ($\geq \text{vecmedian}$) then value in $\text{vecTraining}(j)$ is appended to $\text{onINT}$. Here, expression level greater than median indicates probable gene activation in tumorous case.

```plaintext
% Segregate values based on % status of TRCMPLX
\text{onINn} = []; \\
\text{offINn} = []; \\
\text{onINT} = []; \\
\text{offINT} = [];
for j = 1:lencond
  if labelTraining(j) < 0 & & ...
    \text{vecTraining}(j) < \text{vecmedian}
    \text{offINn} = [\text{offINn}, \text{vecTraining}(j)]; \\
  elseif labelTraining(j) < 0 & & ...
    \text{vecTraining}(j) \geq \text{vecmedian}
    \text{onINn} = [\text{onINn}, \text{vecTraining}(j)]; \\
  elseif labelTraining(j) > 0 & & ...
    \text{vecTraining}(j) < \text{vecmedian}
    \text{offINT} = [\text{offINT}, \text{vecTraining}(j)]; \\
  else
    \text{onINT} = [\text{onINT}, \text{vecTraining}(j)];
end

Before estimating the values for cpt of **DKK1**, it is important to see how (1) the probability table would look like and (2) the probability table is stored in BNT (Murphy et al. 15). Table 9 represents the conditions of Sample as well as TRCMPLX and the probable beliefs of events (**DKK1** being on/off).

With two parents and binary state, the total number of conditions is $2^2$. To estimate the values of the probable beliefs of an event, the following computation is done. The probability of gene expression being active given Sample is normal and TRCMPLX is off i.e $\Pr(\text{DKK1} = \text{Active} | \text{Sample} = \text{Normal}, \text{TRCMPLX} = \text{Off})$, is the fraction of number of 1’s in the normal sample ($a\times p$) and the sum of total number of normal samples ($\alpha$). Similarly, the probability of gene expression being active given Sample is tumorous and TRCMPLX is off i.e $\Pr(\text{DKK1} = \text{active} | \text{Sample} = \text{tumorous}, \text{TRCMPLX} = \text{Off})$, is the fraction of number of 1’s in the tumorous sample ($b\times p$) and the sum of total number of tumorous samples ($\beta$). Complementary conditional probability values for **DKK1** being inactive can easily be computed from the above estimated values.

```plaintext
\begin{verbatim}
\text{Normal} & \text{Off} & h(1) & l(5) \\
\text{Tumorous} & \text{Off} & l(2) & h(6) \\
\text{Normal} & \text{On} & h(3) & l(7) \\
\text{Tumorous} & \text{On} & l(4) & h(8)
\end{verbatim}
Table 9 Conditional probability table for **DKK1** in $M_{PBK}$ (model-t2). h - probability of event being high; l - probability of event being low. Serial numbers in brackets represent the ordering of numbers in vectorial format.

CPT for **DKK1** in $M_{PBK}$ (model-t2)

| Sample | TRCMPLX | $Pr(\text{DKK1=Off})$ | $Pr(\text{DKK1=On})$ |
|--------|---------|------------------------|------------------------|
| Normal | Off     | h(1)                  | l(5)                  |
| Tumorous | Off    | l(2)                  | h(6)                  |
| Normal | On      | h(3)                  | l(7)                  |
| Tumorous | On     | l(4)                  | h(8)                  |

\end{verbatim}

After the values in $T$ has been established, a constant 1 is added as pseudo count to convert the distribution to a prob-

---

18 | 1–25
ability distribution via Dirichlet process. Finally, the frequencies in \( T \) are normalized in order to obtain the final conditional probability values for \( DKK1 \). Estimation of cpts for genes \( SFRP1 \), \( CCND1 \), \( CD44 \), \( WIF1 \), \( MYC \) and \( DKK4 \) which has \( TRCMPLX \) and \( Sample \) as parents require same computations as above. Figure 6 shows the pictorial representation of one of the cpts in \( M_{PBK} \).

\[
\text{Fig. 6 Conditional probability table for node } DKK1 \text{ in } M_{PBK}.
\]

Following the Naive Bayes model presented by Verhaegh et al.\(^2\) and making slight modifications to it, Sinha\(^1\) generated \( M_{NB+MPBK} \). In this all genes have a single parent, namely \( TRCMPLX \) and it is assumed that the predicted state of \( TRCMPLX \) is exactly the same as the quality of the test sample. Thus the initial probability values for \( TRCMPLX \) are assumed to be fixed and no variation is made on it. The conditional probability value for a gene being active or inactive is estimated based on the state of the \( TRCMPLX \).

The segregation of the probability values depends on the following conditions (a) gene is active and \( TRCMPLX \) is on (represented by vector \( \text{onINTrOn} \)) (b) gene is inactive and \( TRCMPLX \) is off (represented by vector \( \text{offINTrOn} \)) (c) gene is active and \( TRCMPLX \) is off (represented by vector \( \text{offINTrOff} \)) and (d) gene is inactive (represented by vector \( \text{onINTrOff} \)). For every \( j^{th} \) entry in the \( \text{vecTraining} \), if the label (\( \text{labelTraining}(j) \)) is \( \leq 0 \) (\( TRCMPLX \) is off) and the \( DKK1 \) gene expression (\( \text{vecTraining}(j) \)) is less than the estimated median (\( \leq \text{vecmedian} \)) then value in \( \text{vecTraining}(j) \) is appended to \( \text{offINTrOff} \). If the label (\( \text{labelTraining}(j) \)) is \( \geq 0 \) (\( TRCMPLX \) is on) and the \( DKK1 \) gene expression (\( \text{vecTraining}(j) \)) is greater than the estimated median (\( \geq \text{vecmedian} \)) then value in \( \text{vecTraining}(j) \) is appended to \( \text{onINTrOn} \). And finally, if the label (\( \text{labelTraining}(j) \)) is \( \geq 0 \) (\( TRCMPLX \) is on) and the \( DKK1 \) gene expression (\( \text{vecTraining}(j) \)) is greater than the estimated median (\( \geq \text{vecmedian} \)) then value in \( \text{vecTraining}(j) \) is appended to \( \text{onINTrOn} \).
Table 10 Conditional probability table for DKK1 in $M_{NB+PBK}$ (model-p1). h - probability of event being high; l - probability of event being low. Serial numbers in brackets represent the ordering of numbers in vectorial format.

| TRCMPLX | Pr(DKK1=Off) | Pr(DKK1=On) |
|---------|--------------|-------------|
| Off     | h (1)        | 1 (3)       |
| On      | h (2)        | 1 (4)       |

Before estimating the values for cpt of DKK1, it is important to see how (1) the probability table would look like and (2) the probability table is stored in BNT (Murphy et al.\textsuperscript{15}). Table 10 represents the conditions of TRCMPLX and the probable beliefs of events (DKK1 being on/off). With a single parent and binary state, the total number of conditions is 2\textsuperscript{1}. To estimate the values of the probability beliefs of an event, the following computation is done. The probability of gene expression being active given TRCMPLX is off i.e Pr(DKK1 = Active | TRCMPLX = Off), is the fraction of number of 1’s in the normal sample (a) and the sum of total number of normal samples (l). Similarly, the probability of gene expression being inactive given TRCMPLX is off i.e Pr(DKK1 = active | TRCMPLX = On), is the fraction of number of 1’s in the tumorous sample (b) and the sum of total number of tumorous samples (Of). Complementary conditional probability values for DKK1 being inactive can easily be computed from the above estimated values. Figure 6 shows the pictorial representation of one of the cpt in $M_{MPBK}$.

```matlab
% Generate frequencies for
% conditional probability values
% pr(DKK1 - On | TRCMPLX - Off)
% # of On's when TRCMPLX is Off
a = length(onINTrOff);
% total # of TRCMPLX is Off
A = length(onINTrOff) + length(offINTrOff);
% pr(DKK1 - On | TRCMPLX - On)
% # of On's when TRCMPLX is On
b = length(onINTrOn);
% total # of TRCMPLX is On
B = length(onINTrOn) + length(offINTrOn);

% Conditional probability table
% for DKK1 given its parents
T = [A-a, a; B-b, b];
[r, c] = size(T);
```

After the values in $T$ has been established, a constant 1 is added as pseudo count to convert the distribution to a probability distribution via Dirichlet process. Finally, the frequencies in $T$ are normalized in order to obtain the final conditional probability values for DKK1. Figure 7 shows the pictorial representation of one of the cpt in $M_{NB+MPBK}$.

```matlab
% Convert the table to probability
% distribution via Dirichlet process
T = T + 1;
for i = 1:r
    T(i,:) = T(i,:)./sum(T(i,:));
end
```

### 2.6.2 DKK2: (model-'t1') Sample is the single parent of DKK2. The conditional probability value for a gene being active or inactive is estimated based on the state of the Sample. The analysis of quality of sample generates frequency estimates that can help derive probability values. These frequencies depict the following cases (a) gene repressed in normal (represented by vector offINn) (b) gene expressed in normal (represented by vector onINn) (c) gene repressed in tumorous (represented by vector offINt) and (d) gene expressed in tumorous (represented by vector onINt), cases. For every $j$th entry in the vecTraining, if the label(labelTraining(j)) is normal ($≤\text{vecmedian}$) then value in vecTraining(j) is appended to offINn. Here, expression level lower than median indicates probable gene repression in normal case. If the label (labelTraining(j)) is normal ($≤\text{vecmedian}$) and the DKK2 gene expression (vecTraining(j)) is greater than the estimated median ($≥\text{vecmedian}$) then value in vecTraining(j) is appended to onINn. Here, expression level lower than median indicates probable gene repression in normal case. If the label (labelTraining(j)) is normal ($≤\text{vecmedian}$) then value in vecTraining(j) is appended to offINt. Here, expression level lower than median indicates probable gene repression in tumour case. And finally, if the label (labelTraining(j)) is tumorous ($≥\text{vecmedian}$) and the DKK2 gene expression (vecTraining(j)) is greater than the estimated median ($≥\text{vecmedian}$) then value in vecTraining(j) is appended to onINt. Here, expression level greater than median indicates probable gene activation in tumorous case.
### Table 11: Conditional probability table for DKK2 in $M_{NB+MPBK}$ (model-t1). h - probability of event being high; l - probability of event being low. Serial numbers in brackets represent the ordering of numbers in vectorial format.

| Sample | $Pr(DKK2=Off)$ | $Pr(DKK2=On)$ |
|--------|----------------|---------------|
| Normal | $l/h$ (1)      | $h/l$ (3)     |
| Tumor  | $h/l$ (2)      | $l/h$ (4)     |

Before estimating the values for cpt of $DKK2$, it is important to see how (1) the probability table would look like and (2) the probability table is stored in BNT (Murphy et al.\textsuperscript{15}).

Table 11 represents the conditions of $Sample$ and the probable beliefs of events ($DKK2$ being on/off). With a single parent and binary state, the total number of conditions is $2^1$. To estimate the values of the probable beliefs of an event, the following computation is done. The probability of gene expression being active given $Sample$ is normal i.e $Pr(DKK1 = $Active $ | Sample = $Normal$)$, is the fraction of number of 1’s in the normal sample ($a$) and the sum of total number of normal samples ($A$). Similarly, the probability of gene expression being active given $Sample$ is tumorous i.e $Pr(DKK2 = $active $ | Sample = $Tumorous$)$, is the fraction of number of 1’s in the tumorous sample ($b$) and the sum of total number of tumorous samples ($B$). Complementary conditional probability values for $DKK2$ being inactive can easily be computed from the above estimated values.

% Generate frequencies for conditional probability values
% # of On’s in Normal
$A = length(onINn) + length(offINn)$;
% # of On’s in Tumor
$B = length(onINT) + length(offINT)$;

After the values in $T$ has been established, a constant 1 is added as pseudo count to convert the distribution to a probability distribution via Dirichlet process. Finally, the frequencies in $T$ are normalized in order to obtain the final conditional probability values for $DKK2$. Estimation of cpts for genes $DKK3 - 1$, $DKK3 - 2$, $SFRP3$ and $LEF1$ which have $Sample$ as parent require same computations as above.

% Conditional probability table for $DKK2$ given its parents
Fig. 8 Conditional probability table for node DKK2 in $M_{PBK+EI}$ and $M_{PBK}$.

$$
\begin{array}{|c|c|c|}
\hline
\text{Sample} & \Pr(\text{DKK2}=\text{Off}) & \Pr(\text{DKK2}=\text{On}) \\
\hline
\text{Normal} & (A-a)/(A+2) & (a+1)/(A+2) \\
\text{Tumor} & (B-b)/(B+2) & (b+1)/(B+2) \\
\hline
\end{array}
$$

2.6.3 DACT3: (model-'t1') The conditional probability value for a gene being active or inactive is estimated from generated frequency estimates that can help derive probability values. These frequencies depict the following cases (a) gene repressed in normal (represented by vector offINn) (b) gene expressed in normal (represented by vector onINn) (c) gene repressed in tumorous (represented by vector offINt) and (d) gene expressed in tumorous (represented by vector onINt), cases. For every $j$th entry in the vecTraining, if the label(labelTraining($j$)) is normal ($\leq 0$) and the $DACT3$ gene expression (vecTraining($j$)) is less than the estimated median ($\leq $vecmedian) then value in vecTraining($j$) is appended to offINn. In case, expression level lower than median indicates probable gene repression in normal case. If the label(labelTraining($j$)) is normal ($\leq 0$) and the $DACT3$ gene expression (vecTraining($j$)) is greater than the estimated median ($\geq $vecmedian) then value in vecTraining($j$) is appended to onINn. Here, expression level greater than median indicates probable gene activation in normal case. If the label(labelTraining($j$)) is tumorous ($\geq 0$) and the $DACT3$ gene expression (vecTraining($j$)) is less than the estimated median ($\leq $vecmedian) then value in vecTraining($j$) is appended to offINt. Here, expression level lower than median indicates probable gene repression in tumorous case. And finally, If the label(labelTraining($j$)) is tumorous ($\geq 0$) and the $DACT3$ gene expression (vecTraining($j$)) is greater than the estimated median ($\geq $vecmedian) then value in vecTraining($j$) is appended to onINt. Here, expression level greater than median indicates probable gene activation in tumorous case.

% Segregate values based on status
% of histone repressive and active
% marks
onINn = []; offINn = []; onINt = []; offINt = [];

for $j = 1:lencond$
    if labelTraining($j$) < 0 && vecTraining($j$) < vecmedian
        offINn = [offINn, vecTraining($j$)];
    elseif labelTraining($j$) < 0 && vecTraining($j$) >= vecmedian
        onINn = [onINn, vecTraining($j$)];
    elseif labelTraining($j$) > 0 && vecTraining($j$) < vecmedian
        onINt = [onINt, vecTraining($j$)];
    else
        offINt = [offINt, vecTraining($j$)];
end
Fig. 9 Conditional probability table for node $DACT3$ in $\mathcal{M}_{PBK+EI}$.

Before estimating the values for cpt of $DACT3$, it is important to see how (1) the probability table would look like and (2) the probability table is stored in BNT (Murphy et al.15). Table 12 represents the conditions of $Sample$, $H3K27me3$ and $H3K4me3$ the probable beliefs of events ($DACT3$ being on/off). Finally, from biological data presented in Jiang et al.3 the conditional probability values for the $DACT3$ gene being active based on the histone modification and the available samples suggest that $DACT3$ expression is high in normal samples when the histone repressive mark $H3K27me3$ is reduced and activating mark $H3K4me3$ are present in high abundance. Thus, the probability i.e $Pr(DACT3 = active|H3K27me3 = low, H3K4me3 = high, Sample = normal)$ is the fraction of the number of 1’s in the normal samples ($a$) and the total number of normal samples ($A$). For all other conditions of $H3K27me3$ and $H3K4me3$ when the $Sample$ is normal the probability of $DACT3$ being active is ($A-a$), i.e flip or complementray of $Pr(DACT3 = active|H3K27me3 = low, H3K4me3 = high, Sample = normal)$. This is because in all other conditions of the histone marks the probability of $DACT3$ being active will be reverse of what it is when $H3K27me3$ is reduced and $H3K4me3$ is present in abundance. Similarly, in case of tumorous samples, the probability of $DACT3$ being active will occur when $H3K27me3$ is reduced and $H3K4me3$ is high abundance (a rare phenomena). Thus the probability i.e $Pr(DACT3 = active|H3K27me3 = low, H3K4me3 = high, Sample = tumorous)$ is the fraction of the number of 1’s in the tumorous sample ($b$) and the total number of tumorous samples ($B$). For all other conditions of $H3K27me3$ and $H3K4me3$ when the $Sample$ is tumorous the probability of $DACT3$ being active is ($B-b$), i.e flip or complementray of $Pr(DACT3 = active|H3K27me3 = low, H3K4me3 = high, Sample = tumorous)$. The reason for flip is the same as described above.

%% Generate frequencies for % conditional probability values % $pr(DACT3 - On | H3K27me3 - 1, H3K4me3 - 2, Sample - Normal)$ % # of $On$’s in $Normal$ a = length(onINn);
% total # of On's in Normal
A = length(offINn) + length(onINn);

% pr(DACT3 - On | H3K27me3 - 1, H3K4me3 - 2, Sample - Tumor)
% # of On's in Tumor
b = length(onINt);

% total # of On's in Tumor
B = length(offINt) + length(onINt);

% In rest of the cases where
% (H3K27me3 - 1 and H3K4me3 - 2) is not
% present, the probabilities reverse.

After the values in T has been established, a constant 1 is
added as pseudo count to convert the distribution to a prob-
ability distribution via Dirichlet process. Finally, the frequen-
cies in T are normalized in order to obtain the final conditional
probability values for DACT3. Figure 9 shows the pictorial
representation of one of the cpt in $M_{PBK+EI}$.

% Conditional probability table
% for DACT3 given its parents
T = [a, A-a;... a, A-a;... A-a, a;... a, A-a;... b, B-b;... b, B-b;... B-b, b;... b, B-b];
[r,c] = size(T);

% Convert the table to probability
% distribution via Dirichlet process
T = T + 1;
for i = 1:r
T(i,:) = T(i,:)./sum(T(i,:));
end

Finally, for every gene, after the computation of the prob-
ability values in their respective cpt, the function generate-
GeneCpd returns the following arguments as output.
gene_cpd = struct();
gene_cpd.vecmedian = vecmedian;
gene_cpd.T = T;

3 A programming project for practice

To get a feel of the project, interested readers might want to
implement the following steps when the evidence evidence is 'me'. The code needs to be embedded as a case in the switch
part of the twoHoldOutExp function. The idea is to perturb
the methylation nodes with binary values and find if one can
converge to the correct prediction of state of TRCMPLX as
well as the Sample. These binary values are stored in a vector
and represents a permutation of the methylation states of the
methylation node in $M_{PBK+EI}$. Varying the values of the
vector can help study how perturbations affect the prediction
of the network and the predictions. The steps are given below-

1. Define variables for storing predictions of TRCMPLX
   (tempTRCMPLX) and Sample (tempSample).

2. Find the total number of methylation cases in
   $M_{PBK+EI}$ and store the number in a variable
   noMethylation.

3. Generate binary values for noMethylation nodes.
   Define a cell (binaryStatesOfMethylation) that
   can store vectors of binary values where every permuta-
   tion represents a set of methylation states. The total num-
   ber of permutations should be $2^{\text{noMethylation}}$ which is
   stored in noMethylationConfig. One might want to use
   quantizer and num2bin functions from matlab.

4. Next, generate methylation evidences. Define a 2D ma-
   trix variable methylationEvidence that stores the
   methylation evidences. One might want to use the mat-
   lab function str2num. Finally, add a value of 1 to methy-
   lationEvidence as the BNT takes in '1' and '2' as states
   representing binary values.

5. Build evidence for inference for every test example. The
   steps following might be necessary
   - For every methylation configuration and for every
     methylation node build evidence.
   - Build a new bayesian network in bnetEngine
     using jtree_inf_engine and store the modi-
     fied engine (in engine) using the function
     enter_evidence.
   - Finally, compute the Pr(TRCMPLX = 2 | ge as ev-
     idence) and Pr(Sample = 2 | ge as evidence) using the
     function marginal_nodes.

6. Store predicted results on observed methylation in struc-
   ture Runs indexed with runCnt.

After the section of new code is filled in, run the code and
check the results.

4 Conclusion

A pedagogical walkthrough of a computational modeling and
simulation project is presented using parts of programming
code interleaved with theory. The purpose behind this endeavor is to acclimatize and ease the understanding of beginner students and researchers in transition, who intend to work on computational signaling biology projects. To this end, static Bayesian network models for the Wnt signaling pathway has been selected for elucidation. This is done due to lack or paucity of manuscripts explaining the computational experiments from tutorial perspective due to restrictive policies.

Acknowledgement

Thanks to - (1) All anonymous reviewers who have helped in refining this manuscript. (2) Netherlands Bioinformatics Centre (NBIC) for funding the project. (3) Dr. ir. R. H. J. Faste- nau (Dean of Faculty of Electrical Engineering Mathematics and Computer Science), Dr. Prof. Ir. K. Ch. A. M. Luy- ben (Rector Magnificus) and Drs. D. J. van den Berg (Presi- dent) at Delft University of Technology, for providing support for conducting this work and giving permission to submit the manuscript. (4) Dr. Wim Verhaegh (faculty at Netherlands Bioinformatics Centre and a principal scientist at Molecular Diagnostic Lab in Philips Research) for providing technical support. (5) Marcel J. T. Reinders (scientific director of Bioinformatics Research at Netherlands Bioinformatics and a professor at Delft University of Technology) for refining the proposed model and suggesting inclusion of methylation data. (6) Dr. Robert P.W. Duin (retired associate professor at Delft University of Technology) for informal discussions on 2-holdout experiments and (7) Dr. Jeroen de Ridder (assistant professor at Delft University of Technology) and (8) Phd candidate Marc Hulsman (Delft University of Technology), for informal discussions on estimation of probability values on methylation data. (9) The author is indebted to Mr. Prabhat Sinha and Mrs. Rita Sinha for financially supporting this project while the author was on educational and work leave.

Conflict of Interest

None.

References

1. S. Sinha, *Integr. Biol.*, 2014, 6, 1034–1048.
2. W. Verhaegh, P. Hatzis, H. Clevers and A. van de Stolpe, *Cancer Research, San Antonio Breast Cancer Symposium*, 2011, 71, 524–525.
3. X. Jiang, J. Tan, J. Li, S. Kivimäe, X. Yang, L. Zhuang, P. Lee, M. Chan, L. Stanton, E. Liu, B. Cheyette and Q. Yu, *Cancer cell*, 2008, 13, 529–541.
4. H. Clevers, *Cell*, 2006, 127, 469–480.
5. A. Gregoirieff and H. Clevers, *Genes & development*, 2005, 19, 877–890.
6. I. Costello and C. Plass, *Journal of medical genetics*, 2001, 38, 285–303.
7. P. Das and R. Singal, *Journal of Clinical Oncology*, 2004, 22, 4632–4642.
8. J. Issa, *Clinical Cancer Research*, 2007, 13, 1634–1637.
9. H. Suzuki, D. Watkins, K. Jair, K. Schuebel, S. Markowitz, W. Chen, T. Pretlow, B. Yang, Y. Akiyama, M. Van Engeland et al., *Nature genetics*, 2004, 36, 417–422.
10. C. Niehrs, *Oncogene*, 2006, 25, 7469–7481.
11. H. Sato, H. Suzuki, M. Toyota, M. Nojima, R. Maruyama, S. Sasaki, H. Takagi, Y. Sogabe, Y. Sasaki, M. Idogawa, T. Sonoda, M. Mori, K. Imai, T. Tokino and Y. Shinomura, *Carecinogenesis*, 2005, 28, 2459–2466.
12. H. Taniguchi, H. Yamamoto, T. Hirata, N. Miyamoto, M. Oki, K. Noshio, Y. Adachi, T. Endo, K. Imai and Y. Shinomura, *Oncogene*, 2005, 24, 7946–7952.
13. B. Strahl and C. Allis, *Nature*, 2000, 403, 41–45.
14. C. Peterson, M. Laniel et al., *Current Biology*, 2004, 14, 546–551.
15. K. Murphy et al., *Computing science and statistics*, 2001, 33, 1024–1034.
16. O. Aguilera, M. Fraga, E. Ballestar, M. Perez, M. Herranz, J. Garcia, A. Munoz, M. Esteller and J. Gonzalez-Sancho, *Oncogene*, 2006, 25, 4116–4121.
17. H. Suzuki, E. Gabrielison, W. Chen, R. Anbazhagan, M. Van Engeland, M. Weijen- berg, B. Schirm and S. Baylin, *Nature genetics*, 2002, 31, 141–149.
18. G. Yuan, C. Wang, C. Ma, N. Chen, Q. Tian, T. Zhang and W. Fu, *PLoS one*, 2012, 7, e34004.
19. L. Kriegel, D. Horst, J. Reinhe, J. Engel, T. Kirchner, A. Jung et al., *J. Transl. Med*, 2010, 8, 123.
20. M. Waterman, *Cancer and Metastasis Reviews*, 2004, 23, 41–52.
21. V. Chui, *PLoS one*, 2011, 6, e18966.
22. K. Schmidt-Ott, T. Masckauchan, X. Chen, B. Hirsh, A. Sarkar, J. Yang, N. Paragas, V. Wallace, D. Dufort, P. Pavlidis et al., *Development*, 2007, 134, 3177–3190.
23. S. Kanwar, Y. Yu, J. Nautiyal, B. Patel and A. Majumdar, *Molecular cancer*, 2010, 9, 212.
24. G. Caldwell, C. Jones, P. Taniere, R. Warren, Y. Soon, G. Matthews and D. Morton, *British journal of cancer*, 2006, 94, 922–927.
25. N. Reguart, B. He, Z. Xu, L. You, A. Lee, J. Mazieres, I. Mikami, S. Batra, R. Rosell, F. McCormick et al., *Biochemical and biophysical research communications*, 2004, 323, 229–234.
26. J. Gonzalez-Sancho, O. Aguilera, J. Garcia, N. Pendás-Franco, C. Peña, S. Cal, A. de Herreros, F. Bonilla and A. Malouf, *Oncogene*, 2004, 24, 1098–1103.
27. N. Pendás-Franco, J. Garcia, C. Peña, N. Valle, M. Palmer, M. Herranz, C. Carlberg, B. Jimenez, F. Bonilla, A. Munoz et al., *Oncogene*, 2008, 27, 4467–4477.
28. S. Baehs, A. Herbst, S. Thieme, C. Perschil, A. Behrens, S. Scheel, A. Jung, T. Brabletz, B. Goke, H. Blum et al., *Cancer letters*, 2007, 276, 152–159.
29. M. Otaka, H. Otsuka, T. Sato et al., *Oncogene*, 2008, 27, 100–109.
30. J. Veem and E. Dahl, *Biochimica et Biophysica Acta (BBA)-Reviews on Cancer*, 2012, 18, 28–28.
31. P. Brophy, K. Lang and G. Dressel, *Journal of Biological Chemistry*, 2003, 278, 52401–52405.
32. Q. Feng Han, W. Zhao, J. Bentel, A. Shearwood, N. Zeps, D. Joseph, B. Iacopetta and A. Dharmanarajan, *Cancer letters*, 2006, 231, 129–137.
33. D. Huang, B. Yu, Y. Deng, W. Sheng, Z. Peng, W. Qin and X. Du, *Journal of cancer research and clinical oncology*, 2010, 136, 395–401.
34. B. Huang, M. Moos, S. Vukicevic and F. Luyten, *Journal of Biological Chemistry*, 2006, 281, 26131–26137.
35. S. Kivimäe, X. Yang and B. Cheyette, *BMC biochemistry*, 2011, 12, 33.
36. E. Charniak, *AI magazine*, 1991, 12, 50.