Supplementary Material

1 Sampling IFs and IFGSs from KEGG for Case Study I

After extracting networks from the KEGG database, IFs and IFGSs are sampled from these networks. The procedures detailed below are for each network.

1.1 Construct a BFS-Forest for each network

This procedure has been modified from the vanilla Depth First Search Algorithm.

Algorithm 1 BFS-Forest

```
1: **Input:** A $d \times d$ adjacency matrix $A$.
2: **Output:** A $d \times d$ BFS-Forest adjacency matrix $F$.
3: Remove all self-transitions in $A$.
4: Find all of the roots of $A$ and store them in a vector $R$.
5: **if** no roots **then**
6:     Sort all vertices in descending order based on their out degree and store them in $R$.
7: **end if**
8: Initialize a $d \times d$ Boolean adjacency matrix $F$ with all entries set to **false**.
9: Initialize a $1 \times d$ vector $nV$ to keep track of the not visited vertices with all elements set to **true**.
10: **for** each vertex $r \in R$ **do**
11:     **if** $nV(r)$ is **true then**
12:         set $nV(r)$ to **false**.
13:         BFS-Visit(r).
14:     **end if**
15: **end for**
16: Return $F$.
```

Algorithm 2 BFS-Visit

```
1: **Input:** A vertex $r$.
2: **Output:** The updated matrix $F$.
3: Initialize a queue $Q$ with the vertex $r$ at its head.
4: **while** $Q$ is not empty **do**
5:     Pop a vertex $v$ from $Q$.
6:     Find all of the neighbors $N$ of $v$.
7:     **if** $N$ is empty **then**
8:         continue
9:     **end if**
10:     **for** each neighbor $n \in N$ **do**
11:         **if** $nV(n)$ is **true then**
12:             set $nV(n)$ to **false**.
13:             Add $n$ to $Q$.
14:             set $F(v, n)$ to **true**.
15:         **end if**
16:     **end for**
17: **end while**
```

1.2 Generating IFs and IFGSs for each network

To sample IFs and IFGSs, Network2GeneSets [1] is run on the BFS-Forest F.

Algorithm 3 Network2GeneSets

```
1: Input: BFS-Forest $F$ with $n$ nodes.
2: Output: IFs and IFGSs in $F$.
3: for $i = 1, \ldots, n$ do
4:   if node $i$ has no children then
5:     continue
6:   else
7:     if node $i$ has children then
8:       add to Queue $Q$ and the Linked List $L$ all the directed pairs consisting of $i$ and a child of $i$
9:     end if
10:    while $Q$ is not empty do
11:      Pop an IF $P$ from $Q$
12:      if the last node in $P$, say $k$, has no children then
13:        continue
14:      end if
15:      add to $Q$ and $L$, all IFs obtained by appending each child of $k$ to $P$
16:    end while
17:  end if
18: end for
19: Prune IFs in $L$ of length $< 4$.
20: Return $L$
21: Return all IFGSs obtained by randomly permuting the order of genes within each IF in $L$, keeping the
    end nodes fixed.
```

2 Hierarchial Arrangement of Genes for Case Study II

In the following tables, we present the hierarchial arrangement of genes present in the ERBB and PMOM pathways in the KEGG database.

	Genes
Layer1	EGF, TGFA, AREG, EREG, BTC, HBGEF,
	ERBB2, NRG1, NRG2, NRG3, NRG4
Layer2	EGFR, ERBB3, ERBB4
Layer3	SRC, CBL, CBLC, CBLB, NCK1, NCK2
	PLCG1, PLCG2, STAT5A, STAT5B, SHC1, SHC2,
	SHC3, SHC4, CRK, CRKL
Layer4	PTK2, PAK1, PAK2, PAK3, PAK4, PAK6, PAK7
	CAMK2A, CAMK2B, CAMK2D, CAMK2G, PRKCB
	PRKCA, PRKCG, GRB2, ABL1,ABL2
Layer5	MAP2K4, MAP2K7, SOS1, SOS2, GAB1
Layer6	MAPK8, MAPK9, MAPK10, NRAS, HRAS, KRAS, PIK3R2
	PIK3CA, PIK3R3, PIK3R5, PIK3CB, PIK3CD, PIK3R1, PIK3CG
Layer7	JUN, ARAF, BRAF, RAF1, AKT1, AKT2, AKT3
Layer8	MAP2K1, MAP2K2, BAD, MTOR, CDKN1A, CDKN1B, GSK3B
Layer9	MAPK1, MAPK3, EIF4EBP1, RPS6KB1, RPS6KB2
Layer10	ELK1, MYC

	Genes
Layer1	HSP90AA1, HSP90AB1, PLK1, SPDYA, SPDYC
Layer2	MOS, CDK2, CDC25A, CDC25B, CDC25C
Layer3	MAP2K1
Layer4	MAPK1, MAPK3
Layer5	RPS6KA1, RPS6KA2, RPS6KA3, RPS6KA6
Layer6	BUB1, PKMYT1
Layer7	MAD1L1, MAD2L1, MAD2L2
Layer8	FZR1
Layer9	ANAPC1, ANAPC2, ANAPC4, ANAPC5, ANAPC7,
	ANAPC10, ANAPC11, ANAPC13, CDC16, CDC23, CDC26, CDC27

Table 1: The hierarchial arrangement of 87 genes from the ERBB signaling pathway (upper) and 35 genes from Progesterone-mediated oocyte maturation pathway (lower) available from KEGG database. These representations can be visualized using Cytoscape [2].

3 Computational Complexity

In this section, we first derive the computational complexity of SA. We then present numerical results comparing the performance of SA and Bayesian network methods in terms of F-score and computational time. Let us write an IFGS compendium as an $m \times n$ matrix, where m is the number of information flow gene sets (IFGSs) and n is the number of distinct genes in the compendium. As described above, if there are k ($k \leq n$) genes in the i^{th} gene set, then the first k locations in the i^{th} row contain non-zero indices representing these genes, and the remaining $n-k$ locations are set to zero. The length of the i^{th} IFGS is the number of non-zero indices in the i^{th} row. If L is the maximum length of IFGSs in the compendium, then the computational complexity of SA in taking a total of J jumps is O(JmL). We can derive the computational complexity of SA from Algorithm 1 presented in the main text.

We start with the computational complexity involved in calculating the energy of a signaling pathway structure. It is the sum of:

1. The computational complexity of estimating the initial probability vector, which is $O(m)$. This is because we only need to count the frequency of genes appearing as the first node among m Markov chains.

2. The computational complexity of estimating the transition probability matrix, which is $O(mL + n) = O(mL)$. Indeed, we first compute the frequency counts of various transitions among m Markov chains, followed by a normalization of each row in the transition matrix. Moreover, $n \leq mL$.

3. The computational complexity involved in computing the likelihood of a Markov chain, which is $O(L)$. For m chains, the complexity is $O(mL)$.

Thus, the computational complexity of calculating the energy of a signaling pathway structure is $O(mL)$.

It can be observed from the pseudo-code in Algorithm 1 that the total computational complexity of SA depends on the following computations:

Outside the loop (Before Step 4)

1. We need to calculate the lengths of IFGSs and the maximum of the lengths. As we only consider non-zero indices in the given matrix, the worst case computational complexity involved in these computations is $O(mL + m) = O(mL)$.

2. At Step 3, we assign random gene orderings to each of the m gene sets and calculate the energy of the resulting structure. The worst case complexity involved in each of these computations is $O(mL)$.

Thus, the total complexity before Step 4 is $O(mL)$.

Inside the loop (Step 4 onwards)

To jump from j^{th} to $(j + 1)^{th}$ network,

1. We need to consider the complexity involved in generating a network from the neighborhood of j^{th} network. Since this requires sampling an index $i \in \{1, \ldots, m\}$ and permuting the order of genes in the i^{th} IFGS, the worst case computational complexity is $O(L)$.

2. We need to consider the complexity involved in calculating the energy of the neighboring network chosen for evaluation, which is $O(mL)$.

Thus, the total computational complexity involved in (1) and (2) above is $O(mL)$, and for a total of J jumps it is $O(JmL)$. As a result, the overall computational complexity outside and inside the loop in Algorithm 1 is $O(mL) + O(JmL) = O(JmL)$.

In the following tables, we present the computational time and performance of SA and two Bayesian network methods K2 and MH using IFGS compendiums of different sizes. Both SA and Bayesian network methods use search strategies for learning multivariate dependencies. Also, both SA and Bayesian network methods infer directed network topologies. Therefore, it is relevant to compare SA and Bayesian network methods in terms of performance and search time.

We use 4 IFGS compendiums among 83 compendiums used in Case Study I. For each algorithm, we list the type of output, computational time and F-Score. As both SA and MH depend on the number of jumps/samples specified by the user, we report the performance of these approaches at iteration $10^3, 10^4, 10^5$ and 2×10^5. We suffix the F-Score (F) and elapsed time (T) accordingly. Since this is not applicable in the case of K2, we report the final values of F (F_{Final}) and T (T_{Final}).

Method	Output Type	F_{10^3}	F_{10^4}	F_{10^5}	$F_{2\times10^5}$ or F_{Final}
SA	Directed	0.57	0.89	1	1
MH-BIC	Directed	0.21	0.27	0.45	0.49
MH-BAYES	Directed	0.11	0.16	0.17	0.21
K2-BIC	Directed	*	*	*	0.41
K2-BAYES	Directed	*	*	*	0.32

Method	Output Type	T_{10^3}	T_{10^4}	T_{10^5}	$T_{2\times10^5}$ or T_{Final}
SA	Directed	0.02	0.18	1.9	3.7
MH-BIC	Directed	0.52	5.1	51.22	103.68
MH-BAYES	Directed	0.49	5.14	53.37	118.06
K2-BIC	Directed	*	*	*	0.07
K2-BAYES	Directed	*	*	*	0.10

Table 2: Comparison of SA and Bayesian network methods in terms of F-Score (upper panel) and computational time (lower panel). We used IFGS compendium with 54 IFGSs. The lengths of IFGSs varied in the range $4-8$. Time is shown in minutes. *Not Applicable

Method	Output Type	F_{10^3}	F_{10^4}	F_{10^5}	$F_{2\times10^5}$ or F_{Final}
SA	Directed	0.69	0.91	1	1
MH-BIC	Directed	0.09	0.22	0.30	0.34
MH-BAYES	Directed	0.08	0.11	-	-
K2-BIC	Directed	*	*	*	0.28
K2-BAYES	Directed	*	*	*	0.20

Method	Output Type	T_{10^3}	T_{10^4}	T_{10^5}	$T_{2\times10^5}$ or T_{Final}
SA	Directed	0.03	0.32	3.2	6.5
MH-BIC	Directed	2.6	25.15	244.15	499.59
MH-BAYES	Directed	2.12	27.02	Out of Memory	Out of Memory
K2-BIC	Directed	*	*	*	0.22
K2-BAYES	Directed	*	*	*	0.27

Table 3: Comparison of SA and Bayesian network methods in terms of F-Score (upper panel) and computational time (lower panel). We used IFGS compendium with 108 IFGSs. The lengths of IFGSs varied in the range $4-7$. Time is shown in minutes. *Not Applicable, $^-$ F-Scores which could not be observed due to memory crash.

Method	Output Type	F_{10^3}	F_{10^4}	F_{10^5}	$F_{2\times10^5}$ or F_{Final}
SA	Directed	0.45	0.54	0.63	0.74
MH-BIC	Directed	0.17	0.39	0.46	0.47
MH-BAYES	Directed	0.09	0.14	-	-
K2-BIC	Directed	*	*	*	0.41
K2-BAYES	Directed	*	*	*	0.37

Method	Output Type	T_{10^3}	T_{10^4}	T_{10^5}	$T_{2\times10^5}$ or T_{Final}
SA	Directed	0.04	0.39	3.9	7.9
MH-BIC	Directed	2.57	24.95	258.28	485.96
MH-BAYES	Directed	2.22	21.11	Out of Memory	Out of Memory
K2-BIC	Directed	*	*	*	0.26
K2-BAYES	Directed	*	*	*	0.32

Table 4: Comparison of SA and Bayesian network methods in terms of F-Score (upper panel) and computational time (lower panel). We used IFGS compendium with 195 IFGSs. The lengths of IFGSs varied in the range $4-10$. Time is shown in minutes. *Not Applicable, $^-$ F-Scores which could not be observed due to memory crash.

Method	Output Type	F_{10^3}	F_{10^4}	F_{10^5}	$F_{2\times10^5}$ or F_{Final}
SA	Directed	0.33	0.48	0.64	0.71
MH-BIC	Directed	0.03	0.11	-	-
MH-BAYES	Directed	0.02	-	-	-
K2-BIC	Directed	*	*	*	0.30
K2-BAYES	Directed	*	*	*	0.24

Method	Output Type	T_{10^3}	T_{10^4}	T_{10^5}	$T_{2\times10^5}$ or T_{Final}
SA	Directed	0.20	2.00	19.91	39.92
MH-BIC	Directed	380.54	2472.71	Too long	Too long
MH-BAYES	Directed	367.52	Out of Memory	Out of Memory	Out of Memory
K2-BIC	Directed	*	*	*	11.45
K2-BAYES	Directed	*	*	*	14.99

Table 5: Comparison of SA and Bayesian network methods in terms of F-Score (upper panel) and computational time (lower panel). We used an IFGS compendium with 723 IFGSs. The lengths of IFGSs varied in the range $4-12$. Time is shown in minutes. *Not Applicable, $^-$ F-Scores which could not be observed due to memory crash or large computational time.

We observe from the above tables that SA benefits from manageable computational complexity and significantly better performance than Bayesian network methods. In particular, SA has a much reduced computational load, both in terms of time and memory requirements, compared with sampling based MH algorithm. Note that both SA and MH depend on the number of jumps/samples specified by the user. However, the complexity of MH is often unmanageable due a large number of neighboring structures of a sampled network. SA only needs to keep track of the best-so-far structure and can be run on a standard desktop.

4 Additional Evaluation Using Gene Sets with Different Pathway Memberships

We further evaluated the performance of SA using IFGS compendiums that comprised of IFGSs with different pathway memberships. In this evaluation, we merged a number of IFGS compendiums derived from different signaling pathway structures into a single compendium. The resulting compendium was given as input to SA. From the output of SA, we divided the inferred IFs into different groups based on the memberships of the corresponding IFGSs in different pathways. We reconstructed signaling pathway structures from the inferred IFs in each group. Finally, we compared the inferred structures with the ones constructed from the true IFs associated with each group. A similar procedure was used in the case of Bayesian network methods. Under the parameter setting of Case Study I, we inferred Bayesian networks using binary discrete data corresponding to IFGS compendiums obtained after merging different compendiums. We divided an inferred network into subnetworks formed by genes present among the IFGSs sharing the same pathway membership. We compared the inferred subnetworks with the ones constructed from the true IFs associated with each group.

We formed three compendiums by merging 5, 10 and 15 IFGS compendiums used in Case Study I. We then evaluated the performance of SA and Bayesian network method on each compendium by following the above procedure. Results from this evaluation have been presented in Figs. 1 and 2. Due to increase in the number of distinct genes and number of IFGSs, it was not feasible to run MH algorithm on the merged compendiums. Therefore, we only present results from the K2 approach. Clearly, we observe a superior performance of SA, both in terms of F-Scores and precision, compared with K2.

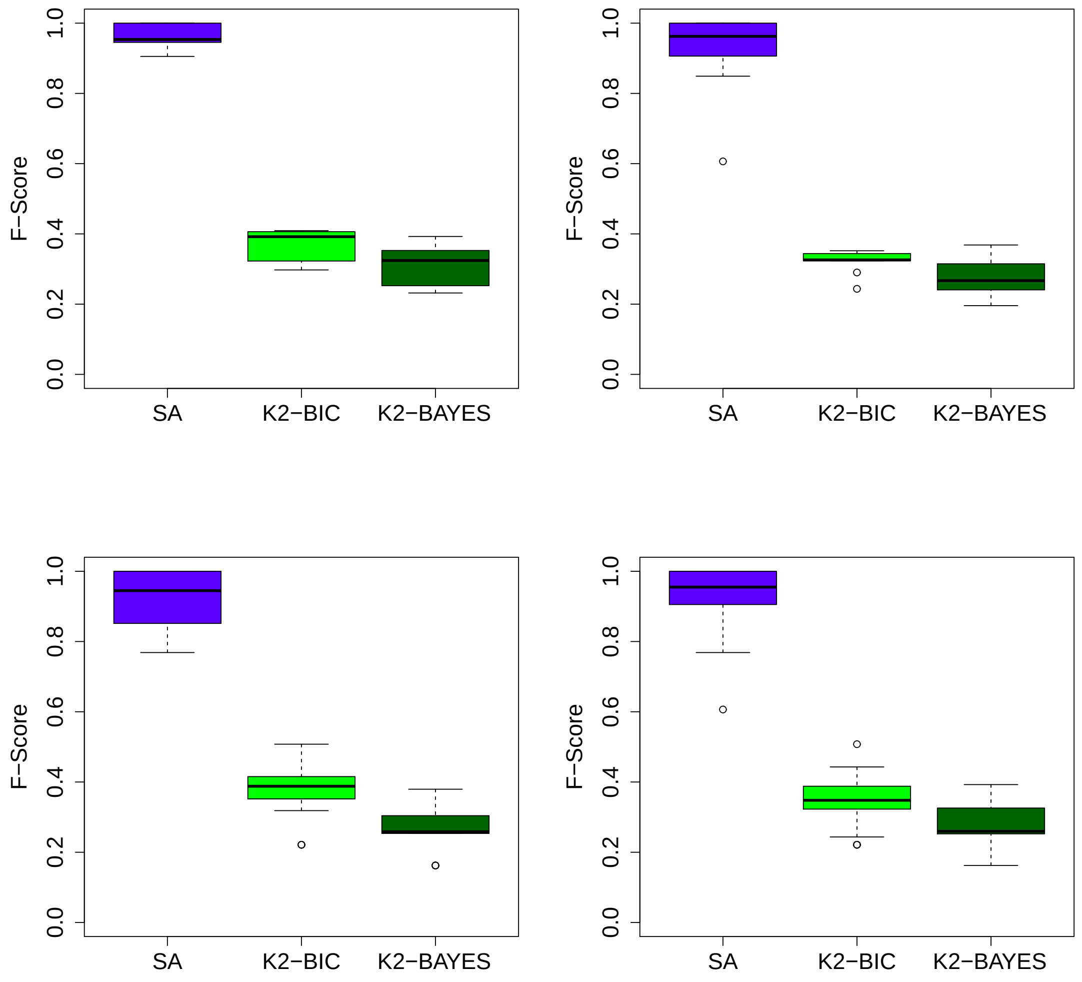

Figure 1: Comparison of SA with the Bayesian network approach K2 using BIC and Bayesian score functions. The number of compendiums merged were set at 5 (Upper Left), 10 (Upper Right) and 15 (Lower Left). The lower right panel presents a combined view. For each method, we plot the F-Score associated with individual compendiums that were merged.

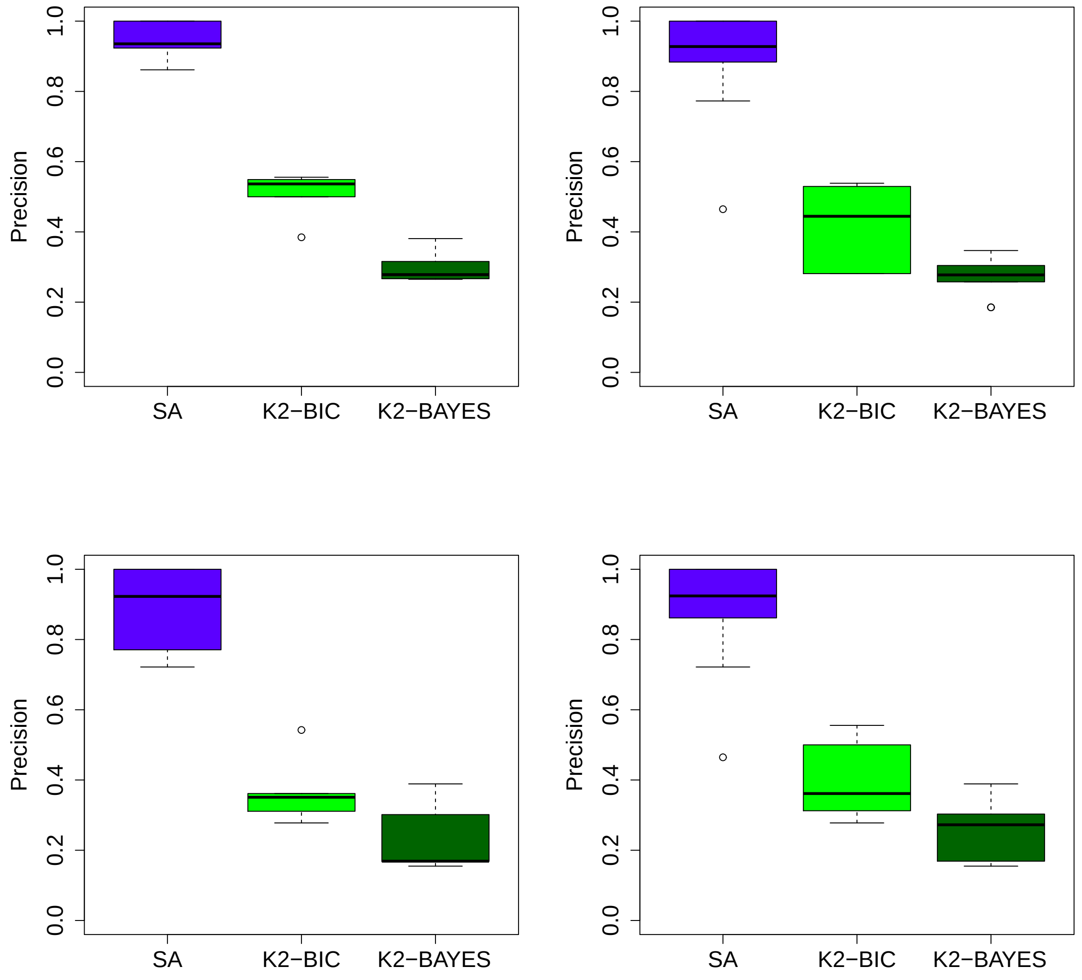

Figure 2: Comparison of SA with the Bayesian network approach K2 using BIC and Bayesian score functions. The number of compendiums merged were set at 5 (Upper Left), 10 (Upper Right) and 15 (Lower Left). The lower right panel presents a combined view. For each method, we plot the precision associated with individual compendiums that were merged.

References

[1] Acharya,L. et al. (2011). GSGS: A computaional framework to reconstruct signaling pathways from gene sets. *arXiv:1101.3983v2* (Preprint).

[2] Shannon, P. et al. (2003) Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Research*, 13(11), 2498-2504.