Algebra for Markov Proposal Kernels

Ian H. Dinwoodie
Duke University

Wogas2
April 2010

Algebra and Sampling

Algebra and Sampling
make a nice couple!

Algebra and Sampling
make a nice couple!

- Contingency tables (linear constraints, nonnegativity): toric ideals; MCMC and SIS
- 0-1 tables (linear constraints, 0-1 valued): Erdos-Gallai, Gale-Ryser Theorem; SIS
- binary sequences in network dynamics (equations not too coupled): elimination ideals; SIS
- graphs, networks (equations highly coupled) : hard

We want to sample from

$$\Omega := L \cap \bigcap_{i=1}^{c} \{g_i(\mathbf{x}) = 0\}$$

Here L is the set of binary or l-level sequences of length d.

We want to sample from

$$\Omega := L \cap \bigcap_{i=1}^{c} \{g_i(\mathbf{x}) = 0\}$$

Here L is the set of binary or l-level sequences of length d.

This is a *fractional design* (Pistone and Rogantin).

($\geq$) Three approaches for sampling from Ω:

($\geq$) Three approaches for sampling from Ω:

1. A Backward SIS method computes elimination ideals over finite fields and constructs partial solutions that extend.

($\geq$) Three approaches for sampling from Ω:

1. A Backward SIS method computes elimination ideals over finite fields and constructs partial solutions that extend.
2. A Forward SIS method uses numerical global optimization to determine which partial solutions extend.

($\geq$) Three approaches for sampling from Ω:

1. A Backward SIS method computes elimination ideals over finite fields and constructs partial solutions that extend.
2. A Forward SIS method uses numerical global optimization to determine which partial solutions extend.
3. Syzygies can be used with MCMC to improve annealing on L.

($\geq$) Three approaches for sampling from Ω:

1. A Backward SIS method computes elimination ideals over finite fields and constructs partial solutions that extend.
2. A Forward SIS method uses numerical global optimization to determine which partial solutions extend.
3. Syzygies can be used with MCMC to improve annealing on L.

Backward Sequential Importance Sampling (BSIS) on Ω:

0. Compute elimination ideals for I_Ω, some polynomials that define Ω and the discrete states.
1. Solve the polynomials in the ideals backwards with random values, like back substitution.
2. A theorem says solutions “extend."
3. Keep track of weights for reweighting.

Example. Aracena (2008) presents an example of network dynamics with a large number of fixed points. Setting $n = 21$ (n being his notation for number of nodes), we have 21 binary maps given by

```
f1=x(2)
f2=x(21)*x(1)
...
...
f17=x(18)
f18=x(21)*x(17)
f19=x(20)
f20=x(21)*x(19)
f21=1-((1-x(2))*(1-x(4))*(1-x(6))*(1-x(8))*(1-x(10))*(1-x(12))*(1-x(14))*
    (1-x(16))*(1-x(18))*(1-x(20)))
```

- We have found $1023 = 2^{(21-1)/2} - 1$ fixed points, not the 1024 that seem to be predicted in Aracena.

Example. Aracena (2008) presents an example of network dynamics with a large number of fixed points. Setting $n = 21$ (n being his notation for number of nodes), we have 21 binary maps given by

```
f1=x(2)
f2=x(21)*x(1)
...
...
f17=x(18)
f18=x(21)*x(17)
f19=x(20)
f20=x(21)*x(19)
f21=1-((1-x(2))*(1-x(4))*(1-x(6))*(1-x(8))*(1-x(10))*(1-x(12))*(1-x(14))*
    (1-x(16))*(1-x(18))*(1-x(20)))
```

- We have found $1023 = 2^{(21-1)/2} - 1$ fixed points, not the 1024 that seem to be predicted in Aracena.
- We can measure the size of the basin of attraction of the fixed point $\mathbf{0}$ – SIS is good for approximate counting! We estimate $|F^{-\infty}(\mathbf{0})| \approx 1 + 1010$, and all points that hit $\mathbf{0}$ do so in 0 or 1 iteration.

- The algebraic BSIS will not handle big problems like social networks.

- The algebraic BSIS will not handle big problems like social networks.
- Forward SIS scales better, it uses global system solvers as a tool to look forward to see which possible current states 0 or 1 will lead to a feasible full sequence.

- The algebraic BSIS will not handle big problems like social networks.
- Forward SIS scales better, it uses global system solvers as a tool to look forward to see which possible current states 0 or 1 will lead to a feasible full sequence.
- The global minimization steps are done numerically with a certain tolerance – runs on large problems with little memory use, but gives samples with some variability in quality.
- Forward SIS can be distributed over many processors.

Forward Sequential Importance Sampling (FSIS) on Ω:

1. Test to see if values 0 or 1 are possible for x_d (last coordinate), by plugging them in and seeing if the dimension $d-1$ equations have *any* solution.

Forward Sequential Importance Sampling (FSIS) on Ω:

1. Test to see if values 0 or 1 are possible for x_d (last coordinate), by plugging them in and seeing if the dimension $d-1$ equations have *any* solution.
2. The way to see if they have a solution is not algebraic this time (*not* is $1 \notin I_\Omega$), rather you convert the problem to global minimization in the usual way, and see if the minimum is 0.

Forward Sequential Importance Sampling (FSIS) on Ω:

1. Test to see if values 0 or 1 are possible for x_d (last coordinate), by plugging them in and seeing if the dimension $d-1$ equations have *any* solution.
2. The way to see if they have a solution is not algebraic this time (*not* is $1 \notin I_\Omega$), rather you convert the problem to global minimization in the usual way, and see if the minimum is 0.
3. Choose the value randomly from the winners, keep track of weights, keep going through $x_d, x_{d-1}, \ldots, x_1$.

Forward Sequential Importance Sampling (FSIS) on Ω:

1. Test to see if values 0 or 1 are possible for x_d (last coordinate), by plugging them in and seeing if the dimension $d-1$ equations have *any* solution.
2. The way to see if they have a solution is not algebraic this time (*not* is $1 \notin I_\Omega$), rather you convert the problem to global minimization in the usual way, and see if the minimum is 0.
3. Choose the value randomly from the winners, keep track of weights, keep going through $x_d, x_{d-1}, \ldots, x_1$.
4. Errors happen.

- Global optimization methods attempt to minimize a real-valued function from any initial point and without convexity assumptions.

- Global optimization methods attempt to minimize a real-valued function from any initial point and without convexity assumptions.
- Nonmonotone line search methods often play a key role but other methods are also possible (Nelder-Mead).

- Global optimization methods attempt to minimize a real-valued function from any initial point and without convexity assumptions.
- Nonmonotone line search methods often play a key role but other methods are also possible (Nelder-Mead).
- We used one by LaCruz, Martinez, and Raydan (2006), a development of the Barzilai-Borwein spectral method, which is refined and implemented in the R package BB (Varadhan and Gilbert, 2008).

Example. Network of mutually “known" researcher in the EIES.1 data set from SIENA.

Example. Network of mutually “known" researcher in the EIES.1 data set from SIENA.

- Want conditional parameter significance. Conditional conclusion for significance agrees with existing method on this example.

Example. Network of mutually “known" researcher in the EIES.1 data set from SIENA.

- Want conditional parameter significance. Conditional conclusion for significance agrees with existing method on this example.
- Example where the ergm software (Handcock, Hunter *et al.*, 2009) has difficulty, and thus one where the conditional approach may be essential, is the network of mutual friends in EIES.1 – could not get fitted parameter values.

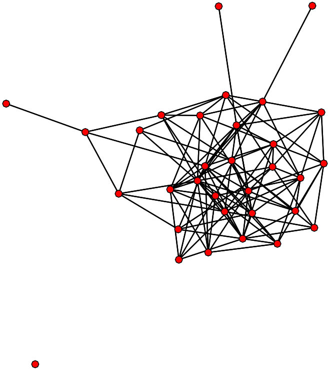

Figure: EIES network of mutually known researchers

If y is a symmetric 0-1 adjacency matrix with no loops, then y_{ij} indicates edge between nodes i and j.

$$E(y) = \sum_{1 \le i < j \le 32} y_{ij}$$

$$T(y) = \sum_{1 \le i < j < h \le 32} y_{ij} y_{ih} y_{jh}$$

$$A(y) = \sum_{2 \le k \le 31} (-1/2)^{k-2} \left(\sum_{i=1}^{32} \binom{y_{i+}}{k} \right)$$

A 3-parameter network probability model is

$$q_{\eta,\tau,\alpha}(y) = \kappa e^{(\eta E(y) + \tau T(y) + \alpha A(y))}$$

We want the significance of the alternating k-star term $A(y)$.

We want the significance of the alternating k-star term $A(y)$.

Fit model with ergm function of the ergm R package:

$\hat{\alpha} = 1.6816$ and reported p-value of 0.630

We want the significance of the alternating k-star term $A(y)$.

Fit model with ergm function of the ergm R package:

$\hat{\alpha} = 1.6816$ and reported p-value of 0.630

- We also want the conditional method: obtain a p-value for $A(y)$ using the conditional distribution of $A(y)$ given the observed value s of $E(y_0)$ and $T(y_0)$.

We want the significance of the alternating k-star term $A(y)$.

Fit model with ergm function of the ergm R package:

$$\hat{\alpha} = 1.6816 \text{ and reported } p\text{-value of } 0.630$$

- We also want the conditional method: obtain a p-value for $A(y)$ using the conditional distribution of $A(y)$ given the observed value s of $E(y_0)$ and $T(y_0)$.
- The conditional distribution is uniform on networks with the same number of edges (113) and triangles (81).

Syzygies for MCMC

$$\pi_\theta(\mathbf{x}) = \frac{e^{-\theta U(\mathbf{x})}}{z_\theta},\ \mathbf{x} \in L$$

where $U := -\sum_{i=1}^{c} g_i^2$.
Metropolis Algorithm on L:

$$K_\theta(\mathbf{x},\mathbf{y}) = K(\mathbf{x},\mathbf{y}) \cdot \min\{1, e^{-\theta(U(\mathbf{y})-U(\mathbf{x}))}\}.$$

Metropolis Algorithm on L:

$$K_\theta(\mathbf{x},\mathbf{y}) = K(\mathbf{x},\mathbf{y}) \cdot \min\{1, e^{-\theta(U(\mathbf{y})-U(\mathbf{x}))}\}.$$

Metropolis Algorithm on L:

$$K_\theta(\mathbf{x},\mathbf{y}) = K(\mathbf{x},\mathbf{y}) \cdot \min\{1, e^{-\theta(U(\mathbf{y})-U(\mathbf{x}))}\}.$$

Some kernels K will be more efficient than others

Metropolis Algorithm on L:

$$K_\theta(\mathbf{x},\mathbf{y}) = K(\mathbf{x},\mathbf{y}) \cdot \min\{1, e^{-\theta(U(\mathbf{y})-U(\mathbf{x}))}\}.$$

Some kernels K will be more efficient than others
in that the proportion of rejected proposal moves will be smaller

Metropolis Algorithm on L:

$$K_\theta(\mathbf{x},\mathbf{y}) = K(\mathbf{x},\mathbf{y}) \cdot \min\{1, e^{-\theta(U(\mathbf{y})-U(\mathbf{x}))}\}.$$

Some kernels K will be more efficient than others
in that the proportion of rejected proposal moves will be smaller
leading to more mobility in the state space, faster convergence
to stationarity.

Let R be the ring of polynomials $\mathbb{Q}[\mathbf{s}] = \mathbb{Q}[s_1, \ldots, s_d]$. Define the gradient $\nabla g_i = (\partial_j g_i)_{j=1,\ldots,d} \in R^d$. Let

$$\partial_j G = \begin{pmatrix} \partial_j g_1 \\ .. \\ .. \\ \partial_j g_c \end{pmatrix}$$

and define the module J to be the span of the polynomial c-tuples $\partial_j G$, with polynomial coefficients $f_j \in \mathbb{Q}[\mathbf{s}]$:

$$J := \{\sum_{j=1}^{d} f_j \cdot \partial_j G\} \in \mathbb{Q}[\mathbf{s}]^c.$$

Consider the syzygy module $S_J \subset R^d$ of d-tuples on the generators $\partial_1 G, \ldots, \partial_d G$ defined by

$$S_J := \{(p_1, \ldots, p_d) \in R^d : p_1 \cdot \partial_1 G + p_2 \cdot \partial_2 G + \cdots + p_d \cdot \partial_d G = 0\}.$$

This can be written in the form

$$\nabla G \cdot P = 0$$

if $P = (p_1, \ldots, p_d)$ is the column of polynomials and G is the derivative matrix

$$\nabla G := \begin{pmatrix} \partial_1 G & \ldots & \partial_d G \end{pmatrix} = \begin{pmatrix} \nabla g_1 \\ \cdots \\ \cdots \\ \nabla g_c \end{pmatrix}.$$

Proposition: Let $\mathbf{x} \in L$ be a particular point, and let a point $\mathbf{y} \in L$ satisfy $\nabla G(\mathbf{x}) \cdot (\mathbf{y} - \mathbf{x}) = 0$. If the matrix $\nabla G(\mathbf{x})$ is of full rank and if the matrix $M_{S_J}(\mathbf{x})$ is of full rank, then $\mathbf{y}$ can be represented as

$$\mathbf{y} = \mathbf{x} + P(\mathbf{x})$$

for some syzygy $P = (p_1, \ldots, p_d) \in S_J$.

Now let M_{S_J} be a $d \times g$ matrix of generators (as columns) for S_J, that is a matrix whose columns are $d \times 1$ vectors of polynomials that are in the module S_J and whose span (with polynomial coefficients) is all of S_J.

$$M_{S_J} := \begin{pmatrix} \mathbf{v}_1 & \cdots & \cdots & \mathbf{v}_g \end{pmatrix}$$

for the $d \times g$ generating matrix of syzygies.

Observe that the acceptance probability $e^{-\theta(U(\mathbf{y})-U(\mathbf{x}))}$ will be on the order of $e^{-\theta\lambda^{\star}\|\mathbf{y}-\mathbf{x}\|^2/2}$ if $\mathbf{y} = \mathbf{x} \pm \mathbf{v}_i(\mathbf{x})$, where $\lambda^{\star}$ is the spectral radius of the second derivative of U at $\mathbf{x}$.

This follows from a Taylor expansion and $\nabla U(\mathbf{x}) \cdot (\mathbf{y} - \mathbf{x}) = 0$.

Observe that the acceptance probability $e^{-\theta(U(\mathbf{y})-U(\mathbf{x}))}$ will be on the order of $e^{-\theta\lambda^\star\|\mathbf{y}-\mathbf{x}\|^2/2}$ if $\mathbf{y} = \mathbf{x} \pm \mathbf{v}_i(\mathbf{x})$, where $\lambda^\star$ is the spectral radius of the second derivative of U at $\mathbf{x}$.
This follows from a Taylor expansion and $\nabla U(\mathbf{x}) \cdot (\mathbf{y} - \mathbf{x}) = 0$.
Since our state space is in the integers, $\|\mathbf{y} - \mathbf{x}\|$ is not necessarily small.

Syzygies as Increments:

- $K_S(\mathbf{x}, \mathbf{y})$ selects a column $\mathbf{v}$ of M_{S_J} uniformly, and adds its randomly-signed evaluation $\sigma\mathbf{v}(\mathbf{x})$ to the current state $\mathbf{x}$.

Syzygies as Increments:

- $K_S(\mathbf{x}, \mathbf{y})$ selects a column $\mathbf{v}$ of M_{S_J} uniformly, and adds its randomly-signed evaluation $\sigma\mathbf{v}(\mathbf{x})$ to the current state $\mathbf{x}$.
- This procedure is not necessarily symmetric, since the increments depend on the state $\mathbf{x}$, and leads to an awkward Metropolis-Hastings algorithm. So a symmetrized version will be used.

Syzygies as Increments:

- $K_S(\mathbf{x},\mathbf{y})$ selects a column $\mathbf{v}$ of M_{S_J} uniformly, and adds its randomly-signed evaluation $\sigma\mathbf{v}(\mathbf{x})$ to the current state $\mathbf{x}$.
- This procedure is not necessarily symmetric, since the increments depend on the state $\mathbf{x}$, and leads to an awkward Metropolis-Hastings algorithm. So a symmetrized version will be used.
- Proposal kernel:

$$K = \frac{1}{2}B_s(\mathbf{x},\mathbf{y}) + \frac{1}{2}K_S(\mathbf{x},\mathbf{y})$$

Example

Symmetric graphs on 4 vertices, with 4 edges and 1 triangle. The adjacency matrices are a subset of binary sequences of length 6, and are written

$$X = \begin{pmatrix} 0 & & & \\ x_1 & 0 & & \\ x_2 & x_3 & 0 & \\ x_4 & x_5 & x_6 & 0 \end{pmatrix}.$$

$$\nabla G = \begin{pmatrix} 1 & \cdots & 1 \\ x_2x_3 + x_4x_5 & \cdots & x_2x_4 + x_3x_5 \end{pmatrix}.$$

Singular gives a set of 11 generators using graded reverse lex order for the syzygies on the Jacobean J. For example, the first one is the column vector

$$(0, -x_2 + x_5, x_3 - x_4, -x_3 + x_4, x_2 - x_5, 0)'.$$

Singular gives a set of 11 generators using graded reverse lex order for the syzygies on the Jacobean J. For example, the first one is the column vector

$$(0, -x_2 + x_5, x_3 - x_4, -x_3 + x_4, x_2 - x_5, 0)'.$$

At a particular state **x** we evaluate:

	1	2	3	4	5	6	7	8	9	10	11
$x_1, 21$	0	0	0	0	1	0	0	0	0	-1	-1
$x_2, 31$	-1	0	-1	-1	-1	0	-1	1	-1	0	2
$x_3, 32$	1	1	1	1	1	0	1	-1	1	0	-2
$x_4, 41$	-1	-1	0	0	-1	0	-1	0	0	1	1
$x_5, 42$	1	0	0	0	0	0	1	0	0	0	0
$x_6, 43$	0	0	0	0	0	0	0	0	0	0	0

Here we see that column 10 added to the present graph will remove edge $\{1, 2\}$ and add edge $\{1, 4\}$, taking us directly from **x** to **y**.

Computation and approximation

A method for cheap syzygies is based on the circuit polynomials. Recall that ∇G is a $c \times d$ matrix and let $c < d$. Consider $c \times d$ indeterminates y_{ij} in a matrix Y:

$$Y = \begin{pmatrix} y_{11} & \cdots & \cdots & y_{1d} \\ \cdots & \cdots & \cdots & \cdots \\ y_{c1} & \cdots & \cdots & y_{cd} \end{pmatrix}.$$

For each subset $C = \{\tau_1, \ldots, \tau_{c+1}\}$ of the $\binom{d}{c+1}$ subsets of size $c+1$ of column indices, form the $d \times 1$ vector $\mathbf{v}_C$ with nonzero entries at coordinates τ_k given by:

$$\mathbf{v}_{C,\tau_k} := (-1)^k \det(Y_{C-\tau_k}),\ k = 1, \ldots, c+1$$

where $Y_{C-\tau_k}$ is the matrix with only the c columns indexed by $C - \{\tau_k\}$. By Cramér's Rule, each vector $\mathbf{v}_C$ is in the kernel of Y with polynomial entries. Now substitute the polynomials $\partial_j g_i(\mathbf{s})$ in for y_{ij} and the result is a syzygy.

Proposition: Let $\mathbf{v}_C(\mathbf{y})$ be the polynomial vector in indeterminates y_{ij} defined above, and let P_C be a d-tuple of polynomials given by $P_C = \mathbf{v}_C(\partial_j g_i(\mathbf{s}))$. Then $\nabla G \cdot P_C = 0$.

Conclusions

- It may be useful to compute syzygies on the columns of the derivative matrix ∇G when trying to sample from a discrete constrained set of the form $G(\mathbf{x}) = 0$.
- The syzygies give a set of tangent vectors that serve as good increments in a Metropolis base chain.
- Theory and examples need more work.