Construction and Random Generation of Hypergraphs with Prescribed Degree and Dimension Sequences

Naheed Anjum Arafat[1], Debabrota Basu[2], Laurent Decreusefond[3], Stéphane Bressan[1]

[1]National University of Singapore
[2]Chalmers University of Technology, Sweden
[3]LTCI, Télécom Paris, France.

November 25, 2020

Hypergraphs

- Data with dyadic relations: Graph (Vertices, Edges)
- Data with poly-adic relations: Hypergraph (Vertices, hyperedges)

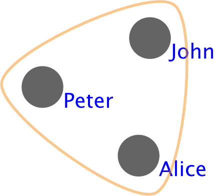


Figure 1: A hypergraph representing a paper co-authored by {John,Alice,Peter}

Hypergraphs

- Data with dyadic relations: Graph (Vertices, Edges)
- Data with poly-adic relations: Hypergraph (Vertices, hyperedges)

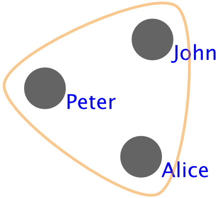


Figure 1: A hypergraph representing a paper co-authored by {John,Alice,Peter}

- **Degree**: $\deg(\text{Alice}) = 1$
- **Dimension**: $\dim(\{John, Alice, Peter\}) = 3$

What is this Talk About?

Generating hypergraphs with a prescribed degree and dimension constraints.

- Peter, Alice, Bob and John wrote 3 , 2, 2 and 2 papers respectively (degree constraint)
- There are three papers with 4, 3 and 2 authors respectively. (dimension constraint)

Generate a hypergraph conforming to these constraints.

Contributions:

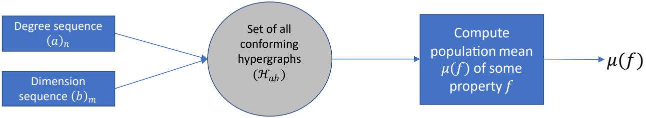
1. An algorithm to construct a conforming hypergraph.
    - Correctness.

Contributions:

1. An algorithm to construct a conforming hypergraph.
    - Correctness.
2. An algorithm to construct a **random** conforming hypergraph.
    - Correctness.
    - Can generate all possible conforming hypergraphs.

Contributions:

1. An algorithm to construct a conforming hypergraph.
   - Correctness.
2. An algorithm to construct a **random** conforming hypergraph.
   - Correctness.
   - Can generate all possible conforming hypergraphs.
3. **Application:**



We devise an Importance Sampling estimator for estimating population mean of some property of the conforming hypergraphs.

Give me a hypergraph: Deterministic Construction.

Input: A Labelled degree sequence and a dimension sequence
Output: A conforming hypergraph (parallel-edges allowed)

Give me a hypergraph: Deterministic Construction.

Input: A Labelled degree sequence and a dimension sequence
Output: A conforming hypergraph (parallel-edges allowed)
The main ideas:

- Construct edges starting from those with the largest to the smallest dimensions.

Give me a hypergraph: Deterministic Construction.

Input: A Labelled degree sequence and a dimension sequence
Output: A conforming hypergraph (parallel-edges allowed)
The main ideas:

- Construct edges starting from those with the largest to the smallest dimensions.
- **Edge construction:** Assign vertices with the largest residual degrees.

	Peter (3)	Alice (2)	Bob (2)	John (2)
e_1 (4)	1	1	1	1
e_2 (3)				
e_3 (2)				

$H = \{\{Peter, Alice, Bob, John\}, \phi, \phi\}$

- Decrease the first |e_1| = 4 components of (3,2,2,2) by 1.
- Sort

	Peter (2)	Alice (1)	Bob (1)	John (1)
e_1 (4)	1	1	1	1
e_2 (3)	1	1	1	
e_3 (2)				

$H = \{$
$\{Peter, Alice, Bob, John\},$
$\{Peter, Alice, Bob\},$
$\phi\}$

- Decrease the first |e_2| = 3 components of (2,1,1,1) by 1.
- Sort

	Peter (1)	Alice (0)	Bob (0)	John (1)
e_1 (4)	1	1	1	1
e_2 (3)	1	1	1	
e_3 (2)	1			1

$H = \{$
$\{Peter, Alice, Bob, John\},$
$\{Peter, Alice, Bob\},$
$\{Peter, John\}\}$

Steps of our proposed algorithm.

Give me a random hypergraph: Random Generation

Generate a random conforming hypergraph.
Requirement: All conforming hypergraphs should be possible to generate.

	Peter (3)	Alice (2)	Bob (2)	John (2)
e_1 (4)	1	1	1	1
e_2 (3)	1	1	1	
e_3 (2)	1			1

$H_1 = \{\{Peter, Alice, Bob, John\}, \{Peter, Alice, Bob\}, \{Peter, John\}\}$

	Peter (3)	Alice (2)	Bob (2)	John (2)
e_1 (4)	1	1	1	1
e_2 (3)	1		1	1
e_3 (2)	1	1		

$H_2 = \{\{Peter, Alice, Bob, John\}, \{Peter, Bob, John\}, \{Peter, Alice\}\}$

	Peter (3)	Alice (2)	Bob (2)	John (2)
e_1 (4)	1	1	1	1
e_2 (3)	1	1		1
e_3 (2)	1		1	

$H_3 = \{\{Peter, Alice, Bob, John\}, \{Peter, Alice, John\}, \{Peter, Bob\}\}$

Give me a random hypergraph: Random Generation

Generate a random conforming hypergraph.
Requirement: All conforming hypergraphs should be possible to generate.

	Peter (3)	Alice (2)	Bob (2)	John (2)
e_1 (4)	1	1	1	1
e_2 (3)	1	1	1	
e_3 (2)	1			1

$H_1 = \{$
$\{Peter, Alice, Bob, John\},$
$\{Peter, Alice, Bob\},$
$\{Peter, John\}\}$

	Peter (3)	Alice (2)	Bob (2)	John (2)
e_1 (4)	1	1	1	1
e_2 (3)	1		1	1
e_3 (2)	1	1		

$H_2 = \{$
$\{Peter, Alice, Bob, John\},$
$\{Peter, Bob, John\},$
$\{Peter, Alice\}\}$

	Peter (3)	Alice (2)	Bob (2)	John (2)
e_1 (4)	1	1	1	1
e_2 (3)	1	1		1
e_3 (2)	1		1	

$H_3 = \{$
$\{Peter, Alice, Bob, John\},$
$\{Peter, Alice, John\},$
$\{Peter, Bob\}\}$

The main idea:

- Construct edges starting from those with the largest to the smallest dimensions.

Give me a random hypergraph: Random Generation

Generate a random conforming hypergraph.
Requirement: All conforming hypergraphs should be possible to generate.

	Peter (3)	Alice (2)	Bob (2)	John (2)
e_1 (4)	1	1	1	1
e_2 (3)	1	1	1	
e_3 (2)	1			1

$H_1 = \{$
$\{Peter, Alice, Bob, John\},$
$\{Peter, Alice, Bob\},$
$\{Peter, John\}\}$

	Peter (3)	Alice (2)	Bob (2)	John (2)
e_1 (4)	1	1	1	1
e_2 (3)	1		1	1
e_3 (2)	1	1		

$H_2 = \{$
$\{Peter, Alice, Bob, John\},$
$\{Peter, Bob, John\},$
$\{Peter, Alice\}\}$

	Peter (3)	Alice (2)	Bob (2)	John (2)
e_1 (4)	1	1	1	1
e_2 (3)	1	1		1
e_3 (2)	1		1	

$H_3 = \{$
$\{Peter, Alice, Bob, John\},$
$\{Peter, Alice, John\},$
$\{Peter, Bob\}\}$

The main idea:

- Construct edges starting from those with the largest to the smallest dimensions.
- **Edge construction:** At each iteration,
  - Divide the columns with non-zero degrees into blocks of intervals.
  - Determine the number of vertices to select from each block.
  - Select the required # of vertices from each block uniformly at random.

Random Edge construction

Suppose, the algorithm already constructed edge $e_1 = \{Peter, Alice, Bob, John\}$.

Residual dim. sequence: $(3, 2)$.
Residual deg. sequence: $(2, 1, 1, 1)$

	Peter (2)	Alice (1)	Bob (1)	John (1)
e_1 (4)	1	1	1	1
e_2 (3)				
e_3 (2)				

Random Edge construction

Suppose, the algorithm already constructed edge $e_1 = \{Peter, Alice, Bob, John\}$.

Residual dim. sequence: $(3, 2)$.
Residual deg. sequence: $(2, 1, 1, 1)$

	Peter (2)	Alice (1)	Bob (1)	John (1)
e_1 (4)	1	1	1	1
e_2 (3)				
e_3 (2)				

Constructing e_2:

- Construct conjugate of the residual dimension sequence (ignoring first component):
  - Conjugate of $(\_, 2) = (\#\text{integers} >= 1, \#\text{integers} >= 2, ..) = (1, 1, 0, 0)$.

Random Edge construction

Suppose, the algorithm already constructed edge $e_1 = \{Peter, Alice, Bob, John\}$.

Residual dim. sequence: $(3, 2)$.
Residual deg. sequence: $(2, 1, 1, 1)$

	Peter (2)	Alice (1)	Bob (1)	John (1)
e_1 (4)	1	1	1	1
e_2 (3)				
e_3 (2)				

Constructing e_2:

- Construct conjugate of the residual dimension sequence (ignoring first component):
  - Conjugate of $(\_, 2) = (\#\text{integers} >= 1, \#\text{integers} >= 2, ..) = (1, 1, 0, 0)$.
  - Construct its prefix sums : $(1, 2, 2, 2)$ (Say, (β))

Random Edge construction

Suppose, the algorithm already constructed edge $e_1 = \{Peter, Alice, Bob, John\}$.

Residual dim. sequence: $(3, 2)$.
Residual deg. sequence: $(2, 1, 1, 1)$

	Peter (2)	Alice (1)	Bob (1)	John (1)
e_1 (4)	1	1	1	1
e_2 (3)				
e_3 (2)				

Constructing e_2:

- Construct conjugate of the residual dimension sequence (ignoring first component):
  - Conjugate of $(\_, 2) = (\#\text{integers} >= 1, \#\text{integers} >= 2, ..) = (1, 1, 0, 0)$.
  - Construct its prefix sums : $(1, 2, 2, 2)$ (Say, (β))
- Construct prefix sums of the residual degree sequence: $(2, 3, 4, 5)$ (Say, (α))

Random Edge construction

Suppose, the algorithm already constructed edge $e_1 = \{Peter, Alice, Bob, John\}$.

Residual dim. sequence: $(3, 2)$.
Residual deg. sequence: $(2, 1, 1, 1)$

	Peter (2)	Alice (1)	Bob (1)	John (1)
e_1 (4)	1	1	1	1
e_2 (3)				
e_3 (2)				

Constructing e_2:

- Construct conjugate of the residual dimension sequence (ignoring first component):
    - Conjugate of $(\_, 2) = (\#\text{integers} >= 1, \#\text{integers} >= 2, ..) = (1, 1, 0, 0)$.
    - Construct its prefix sums : $(1, 2, 2, 2)$ (Say, (β))
- Construct prefix sums of the residual degree sequence: $(2, 3, 4, 5)$ (Say, (α))
- Compute $(\alpha) - (\beta) : (1, 1, 2, 3)$

Random Edge construction

Suppose, the algorithm already constructed edge $e_1 = \{Peter, Alice, Bob, John\}$.

Residual dim. sequence: $(3, 2)$.
Residual deg. sequence: $(2, 1, 1, 1)$

	Peter (2)	Alice (1)	Bob (1)	John (1)
e_1 (4)	1	1	1	1
e_2 (3)				
e_3 (2)				

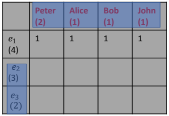
Constructing e_2:

- Construct conjugate of the residual dimension sequence (ignoring first component):
  - Conjugate of $(\_, 2) = (\#\text{integers} >= 1, \#\text{integers} >= 2, ..) = (1, 1, 0, 0)$.
  - Construct its prefix sums : $(1, 2, 2, 2)$ (Say, (β))
- Construct prefix sums of the residual degree sequence: $(2, 3, 4, 5)$ (Say, (α))
- Compute $(\alpha) - (\beta) : (1, 1, 2, 3)$
  - **At least 1,1,2 and 3 vertices need to be selected from the first column, first 2 columns, first 3 columns and first 4 columns resp.** (Dominance conditions.)

Random Edge construction

Suppose, the algorithm already constructed edge $e_1 = \{Peter, Alice, Bob, John\}$.

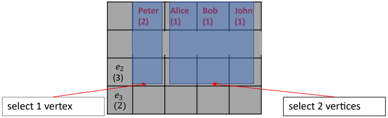
Residual dim. sequence: $(3, 2)$.
Residual deg. sequence: $(2, 1, 1, 1)$



Constructing e_2:

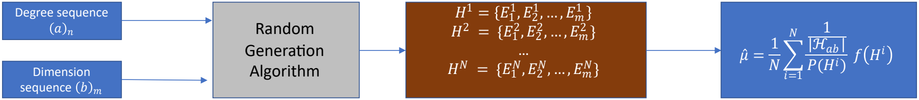
- Construct conjugate of the residual dimension sequence (ignoring first component):
  - Conjugate of $(\_, 2) = (\#\text{integers} >= 1, \#\text{integers} >= 2, ..) = (1, 1, 0, 0)$.
  - Construct its prefix sums : $(1, 2, 2, 2)$ (Say, (β))
- Construct prefix sums of the residual degree sequence: $(2, 3, 4, 5)$ (Say, (α))
- Compute $(\alpha) - (\beta) : (1, 1, 2, 3)$
  - **At least 1,1,2 and 3 vertices need to be selected from the first column, first 2 columns, first 3 columns and first 4 columns resp.** (Dominance conditions.)

- Divide the columns into disjoint blocks such that the dominance conditions are satisfied.



Application: Property estimation

Suppose, we are interested in some property $f : \mathcal{H}_{ab} \to \mathbb{R}$ of the conforming hypergraphs of a degree sequence $(a)_n$ and dimension sequence $(b)_m$



$|\mathcal{H}_{ab}|$ = number of conforming hypergraphs
$P(H^i)$ = Probability of generating hypergraph H^i

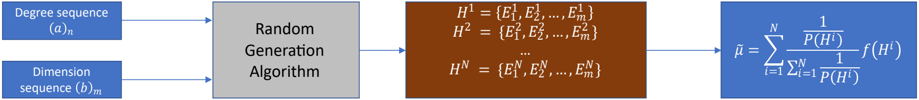
Importance sampling Estimator of $E[f]$.

- **Good**:
  - The probabilities can be computed during random generation.
  - $\hat{\mu}$ is an unbiased estimator of $E[f]$ [1]
- **Bad**: $|\mathcal{H}_{ab}|$ is often unknown.

[1](Proof: Extended paper arXiv:2004.05429)

Application: Property estimation

A practical solution: Normalise the weights in the summation.



Self-Normalised Importance Sampling (SNIS) Estimator for $E[f]$.

We use $\tilde{\mu}$ to estimate population mean $E(f)$.
f =*average clustering coefficient of the graph of the conforming hypergraphs*.

Experimental Results

Datasets:

Datasets	|Degree seq.|	|Dimension seq.|
G_1	6	9
G_2	15	27
G_3	42	81
G_4	123	243
G_5	366	719
G_6	1095	2187
Enron	4423	15653
Congress_bills	1718	260851

Competing Algorithm. MCMC (Philip S. Chodrow, 2019, arXiv)

- Start with an initial conforming hypergraph.
- For T iterations,
    - select a pair of random edges
    - Exchange vertices to produce new edge-pairs with some prob.

Effectiveness Comparison

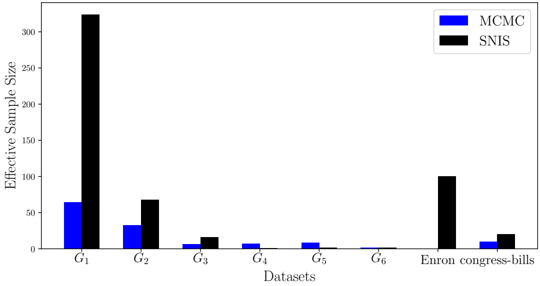


Figure 2: Effective sample sizes of SNIS and MCMC algorithms on different datasets. Higher effective sample size indicates better quality of samples.

- ESS represents the number of i.i.d samples which have the same accuracy as the Monte Carlo estimates (e.g. SNIS, MCMC estimates)
- SNIS estimates equates to more i.i.d sample than MCMC estimates in most of the datasets.

Efficiency Comparison

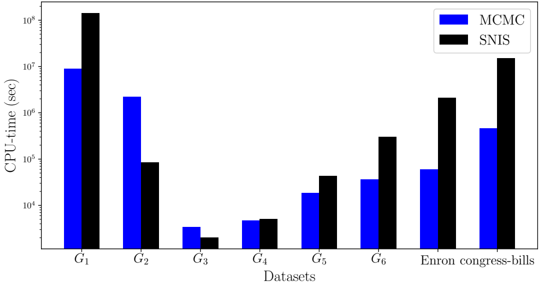


Figure 3: CPU-time (in second, log-scale) to generate 500 hypergraphs for G_1-G_6, 100 hypergraphs for Enron and 20 hypergraphs for congress-bills datasets.

- Efficiency of the random generation algorithm is an issue.
- **Silver lining:** It is often easier to parallelize a direct generation algorithm than an Markov chain based generation algorithm.

Summary.

- We propose an algorithm that constructs a hypergraph with a prescribed degree and dimension sequence.
- We propose an algorithm that generates a random hypergraph with a prescribed degree and dimension sequence.
- We propose an SNIS estimator to estimate hypergraph properties and use it to empirically evaluate the effectiveness of random generation.
  - SNIS estimator has higher effective sample sizes compared to the MCMC estimator.

Summary.

- We propose an algorithm that constructs a hypergraph with a prescribed degree and dimension sequence.
- We propose an algorithm that generates a random hypergraph with a prescribed degree and dimension sequence.
- We propose an SNIS estimator to estimate hypergraph properties and use it to empirically evaluate the effectiveness of random generation.
  - SNIS estimator has higher effective sample sizes compared to the MCMC estimator. '
- **Issue:** Efficiency of the random generation.
  - **Future work:** Random generation in parallel is underway.

Q&A

Supplementary slide

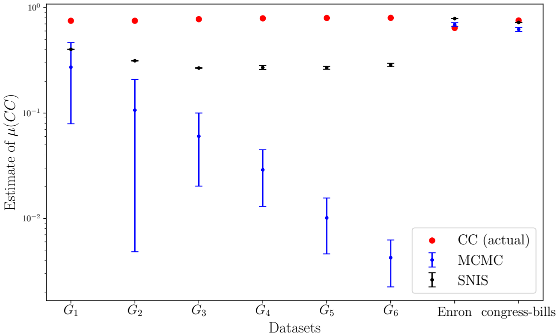


Figure 4: Average clustering coefficients (in log-scale) of the projected random hypergraphs of different datasets and corresponding estimates $\mu(CC)$ using SNIS and MCMC algorithms.

SNIS estimates are more accurate and stable than the MCMC estimates.

Supplementary slide

Gale-Rysers criteria:

An incidence matrix with column-sum $(a)_n$ and row-sum $(b)_m$ exists if and only if $(a)_n$ is dominated by the conjugate of $(b)_m$

Example:

Is there any conforming hypergraph with degrees $(3, 2, 2, 2)$ and dimensions $(4, 3, 2)$?

- Conjugate of $(4, 3, 2) = (\#\text{components} \geq 1, \#\text{components} \geq 2, \#\text{components} \geq 3, \ldots) = (3, 3, 2, 1)$
- Dominance check:
  - Is $a_1 \leq \bar{b}_1$? : $(3 \leq 3)$ $\rightarrow$yes
  - Is $a_1 + a_2 \leq \bar{b}_1 + \bar{b}_2$? : $(3 + 2 \leq 3 + 3)$? $\rightarrow$yes
  - Is $a_1 + a_2 + a_3 \leq \bar{b}_1 + \bar{b}_2 + \bar{b}_3$? : $(3 + 2 + 2 \leq 3 + 3 + 2)$? $\rightarrow$yes
  - Is $a_1 + a_2 + a_3 + a_4 == \bar{b}_1 + \bar{b}_2 + \bar{b}_3 + \bar{b}_4$? : $(3 + 2 + 2 + 2 == 3 + 3 + 2 + 1)$?:yes

Supplementary slide

	Any $(a)_n$, $(b)_m = (2, \ldots, 2)$ (graph)	Any $(a)_n$, $(b)_m = (k, \ldots, k)$ (k-uniform hyp.)	Any $(a)_n$ and $(b)_m$ (hyp.)
Simple	Erdös-Gallai: $\sum a_i$ even and $\forall k$, $\sum_{i=1}^{k} a_i \leq k(k-1) + \sum_{i=k+1}^{n} \min(a_i, k)$	Unknown	Unknown
Parallel-edge allowed	Hakimi: $\sum a_i$ even and $a_1 \leq a_2 + \ldots + a_n$	Billington: $\sum a_i = km$ and $\forall i,\ a_i \leq m$	Gale-Ryser: $\sum a_i = \sum b_i$ and $\forall k$, $\sum_{i=1}^{k} a_i \leq \sum_{i=1}^{k} b_i^*$

Table 1: Results on necessary and sufficient conditions for sequences to be realisable as graphs and hypergraphs. Degree sequence is (a) and dimension sequence (b).