Update

Anthony Ebert

07/10/2019

Update 2019-10-07

Last week we spoke about whether the base measure h is important, and whether we are working on the space of networks, or the space of sufficient statistics and whether it matters. I thought the base measure was arbitrary, but last week I reported that, for ERGMS, h is 1 when we are working on the sample space (space of networks) and equal to the number of combinations of networks corresponding to a particular set of summary statistics when working on the space of summary statistics.

I've thought about this more now, and I have concluded that there are separate KL divergences depending on whether we are working on the sample space (space of networks) or the space of summary statistics.

The advantage of working on the sample space is that we don't need to compute h, however estimating entropy requires a large sample size. The advantage of working on the sample space is that fewer samples are required to accurately estimate the entropy, however h must be computed. This problem is non-trivial and mentioned in very few papers.

My previous results, for the Erdös-Renyi graph, are computed on the summary statistics rather than the original sample space of networks. This was possible because the expression for h is known for Erdös-Renyi graphs.

I adapt the entropy function to work on the sample space itself, by assigning a unique id (hash) to each unique adjacency matrix. I then move to the space of permuted networks (networks which are equivalent with some node relabelling) — it turns out that two adjacency matricies with identical eigenvalues are equivalent graphs (up to node relabelling).

On the sample space

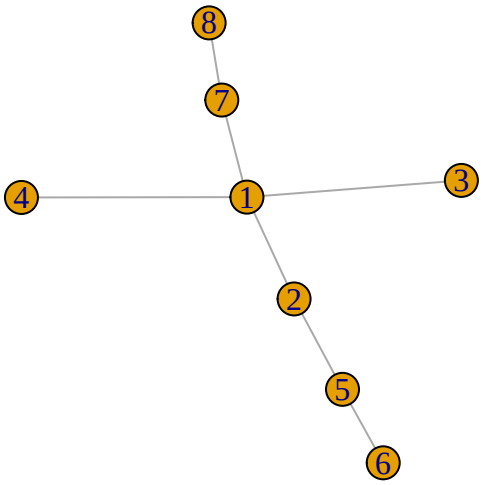
$$\begin{aligned} KL[q(\cdot)||p(\cdot|\theta)]_{\mathcal{Y}} &= \sum_{y\in\mathcal{Y}} q(y)\log\frac{q(y)}{p(y|\theta)} \\ &= \sum_{y\in\mathcal{Y}} q(y)\log q(y) + \log z(\theta) - E_{y\sim q}[\theta' S(y)] \end{aligned}$$

On a space of summary statistics

$$\begin{aligned} KL[q(\cdot)||p(\cdot|\theta)]_{\mathcal{S}(\mathcal{Y})} &= \sum_{s_y\in\mathcal{S}(\mathcal{Y})} q_S(y)\log\frac{q_S(y)}{p(s_y|\theta)} \\ &= \sum_{s_y\in\mathcal{S}(\mathcal{Y})} q_S(s_y)\log q_S(s_y) + \log z(\theta) - E_{s_y\sim q_S}[\theta' s_y] - E_{s_y\sim q_S}[\log h(s_y)] \end{aligned}$$

```
library(igraph)
library(purrr)
library(digest)
library(StartNetwork)
```

```
set.seed(1)

# Barabasi Albert model

x <- replicate(1000, sample_pa(8, directed = FALSE), simplify = FALSE)

plot(x[[1]])
```




```
y <- sapply(x, purrr::compose(digest, igraph::as_adj))

head(y)
```

```
## [1] "28089a521db215979efa78af46501d2b" "74346bca0a0d517005cc5f57b2591e4a"
## [3] "e3cec67ee58fd427c8f13a9a22d5927b" "0e25789b9a7371a130eac362a4748f27"
## [5] "8b440449bbdea45aa28849c2fb212fc7" "2a93f2bd94fd074d0a8c441573ea519c"
```

```
entropy_calc(y)
```

```
## [1] 8.079798
```

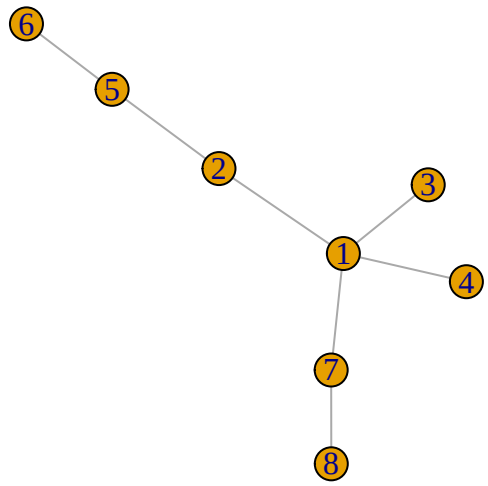
```
y_permute <-
  sapply(x,
    purrr::compose(digest, partial(round, digits = 5), sort, ~ .$values, eigen, igraph::as_adj))

entropy_calc(y_permute)
```

```
## [1] 3.574619
```

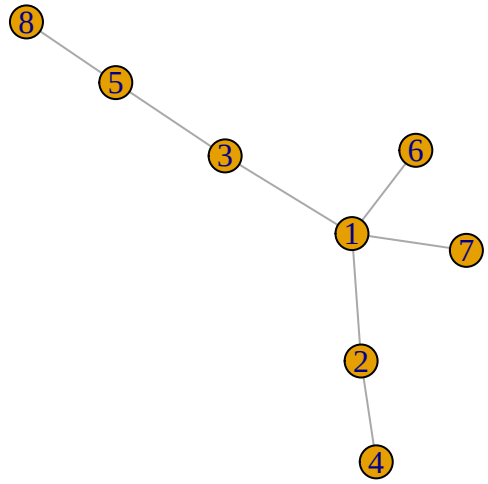
```
equivalent <- which(!nonduplicated(y_permute))[1]

which(y_permute == y_permute[equivalent])
```

```
##  [1]   1 166 171 176 188 207 223 237 357 377 383 398 415 486 536 583 587
## [18] 614 657 659 701 823 897 912 951 986
```

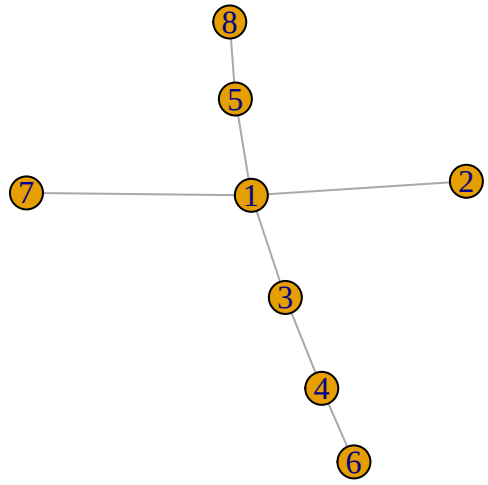
```
plot(x[[1]], label=1:8)
```




```
plot(x[[166]], label=1:8)
```




```
plot(x[[171]], label=1:8)
```



```r
library(ergm)
```

```r
set.seed(2)

x <- simulate(network(6, directed = FALSE) ~ edges + triangle, coef = c(0, 0.01), nsim = 2000)

entropy_calc(x, hash = TRUE)
```

```
## [1] 10.28286
```

```r
y_permute <- sapply(x, purrr::compose(digest, partial(round, digits = 5), sort, ~ .$values, eigen, netw
```

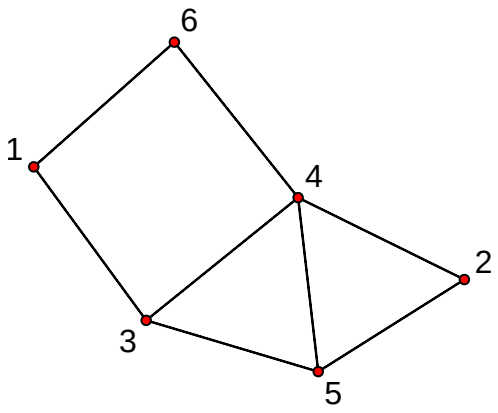
```r
entropy_calc(y_permute)
```

```
## [1] 5.008867
```

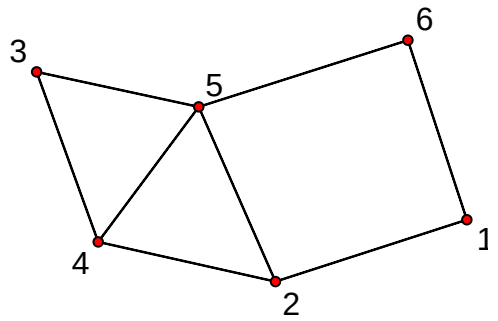
```r
equivalent <- which(!nonduplicated(y_permute))[1]

which(y_permute == y_permute[equivalent])
```

```
##  [1]    1   14   44   86  140  151  299  340  346  352  469  587  593  741
## [15]  776  779  786  824  897  921  951  981 1003 1049 1076 1436 1475 1480
## [29] 1513 1650 1711 1714 1724 1777
```

```r
plot(x[[1]], label=1:6)
```




```r
plot(x[[14]], label=1:6)
```



```
plot(x[[44]], label=1:6)
```