Guide to Calculating Textual Spanning in R

July 14th, 2020

Dustin Stoltz and Marshall Taylor

Textual Spanning

Textual spanning (https://github.com/dustinstoltz/textSpan) is a measure of non-redundance that is specifically adapted to the unique properties of text networks: (1) they are full-connected, undirected, and weighted graphs, (2) and edge strength (generally, a measure of document similarity) is not a finite or transitive quantity. The measure builds on Burt's constraint (Burt 1992), which is a measure of dependence in social networks defined as:

$$C_i = \sum_j \left(p_{ij} + \sum_q p_{qj} p_{iq} \right)^2$$

The primary difference between the Burt's measure of constraint and our measure of spanning is: instead of multiplying p_{qj} by p_{iq}, we divide the two quantities. There are two reasons why this matters for text networks: (1) edge strength is not a finite resource (unlike social networks where we only have so much time and energy to put into our social relations), and (2) we want to reward documents if they have a high similarity to two documents which have low similarities to each other (and vice versa). Formally, we define a document's cumulative textual spanning S_i as:

$$S_i = \sum_j \left(p_{ij} + \sum_q \frac{p_{qj}}{p_{iq}} \right)^2$$

Where p_{ij} is the similarity between document i and document j divided by i's degree (i.e., the sum of i's similarities to all other documents q). This is the second difference with Burt's constraint. Following Opsahl et al. (2010), we divide a_{ij} (or the raw similarity between i and j) by a modified version of weighted degree centrality using a tuning parameter α to stress tie weighting over tie numbers. Therefore, we define proportional similarities p_{ij} as

$$p_{ij} = \frac{a_{ij}}{k_i \times \left(\frac{\sum_q a_{iq}}{k_i} \right)^{\alpha}}$$

Here, k is the count of q vertices adjacent to i. This is multiplied by the sum of a_{iq}, or the weighted degree centrality (i.e., total similarity between all of i's adjacent documents q), divided by k. The tuning parameter α determines the relative importance of the number of total ties k compared to the weights of those ties a_{iq}. (Setting $\alpha = 1$ makes p_{ij} equal a_{ij} divided by the sum of a_{iq}.)

The second term in the first equation, $\frac{p_{qj}}{p_{iq}}$, is the proportional similarity of any third $Document_q$ with $Document_j$ divided by the proportional similarity of $Document_i$ to q which will penalize i's spanning score if the top term is high relative to the bottom.

Finally, to make the measure more interpretable, we standardize the output by taking the z-score of each S_i and inverting it such that positive values indicate more textual spanning:

$$z(S_i) = \left(\frac{s_i - \bar{s}}{\sqrt{\frac{\sum_{i=1}^{n}(s-\bar{s})^2}{n-1}}} \right) \times -1$$

Step by Step Example

```
# packages we'll be using
library(igraph)
library(reshape2)
library(ggraph)
library(knitr)
library(gridExtra)
library(ggplot2)
library(ggpubr)
```

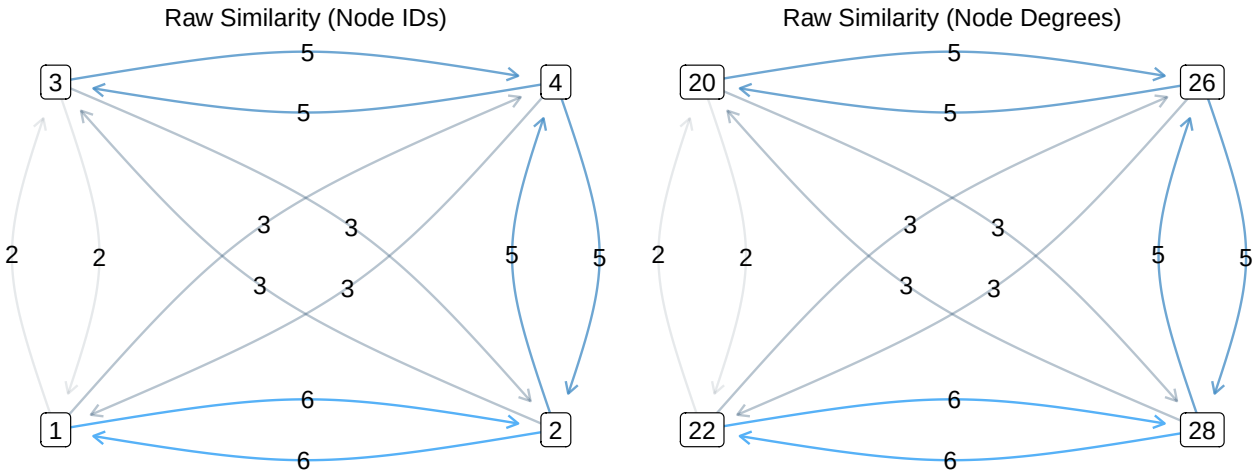
Build and Prepare Hypothetical Text Network

We build the matrix for a hypothetical text network of 5 documents, $matrix_a$. The numbers in each cell refer to some kind of similarity between the $document_i$ and $document_j$. Since we are using integers to start, and for simplicity, let's just say these are a raw count of words shared by each pair of documents. We also zero the diagonals (where documents refer to themselves) so as to ignore them.

```
# create 5 by 5 square matrix with random numbers from 1 to 9 in each cell
n <- 4
set.seed(1618)
mat_a <- matrix(sample.int(6, n*n, TRUE), n, n)
# mat_a <- matrix(sample.int(2, n*n, TRUE), n, n)
mat_a[lower.tri(mat_a)] <- t(mat_a)[lower.tri(mat_a)]
diag(mat_a) <- 0
```

	1	2	3	4
1	0	6	2	3
2	6	0	3	5
3	2	3	0	5
4	3	5	5	0

We can then treat this document-by-document similarity matrix as an adjacency matrix and look at this as a graph to get a better sense of what textual spanning is measuring:



Look at $Document_4$, it has a similarity of 5 with $Document_2$ and with $Document_3$. $Document_2$ and $Document_3$, however, only have a similarity of 3. In this triad, therefore, $Document_4$ is *spanning* a kind of discursive hole between $Document_2$ and $Document_3$ – textual spanning will reward $Document_4$ for this relationship. However, as $Document_4$ is fairly similar to $Document_2$, and $Document_1$ is even more similar to $Document_2$ – textual spanning will also penalize $Document_4$ for this (moderate) similarity to $Document_1$. The output balances out these redundant and non-redundant similarities to arrive at an aggregate spanning score. In this text network, we expect $Document_4$ and $Document_2$ to be the highest spanners.

Calculating Textual Spanning

Proportional Similarities

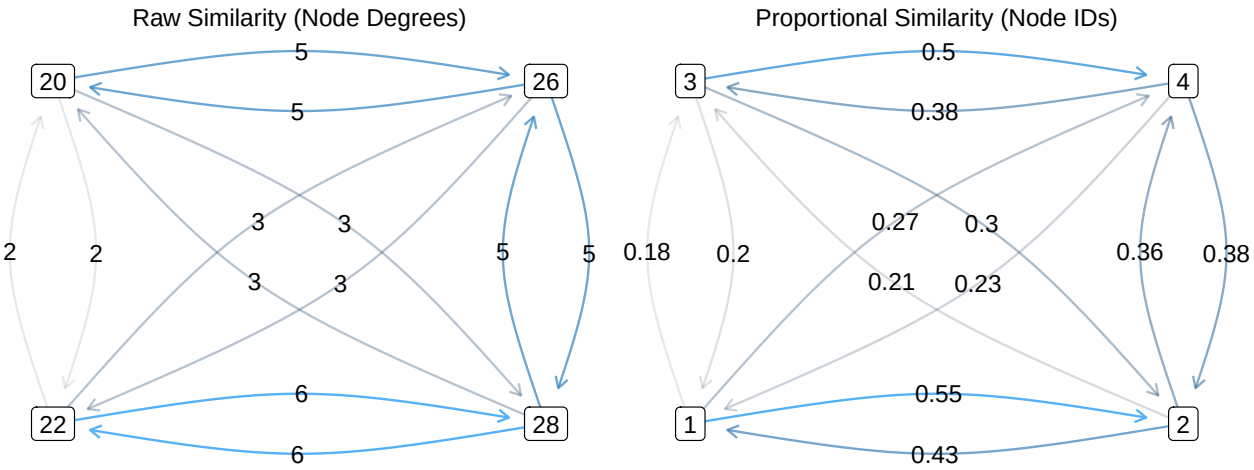
The first step is essentially "normalizing" the text network by dividing each element of $Document_i$ by the sum of non-zero similarities between $Document_i$ and all other documents $Document_q$.

In nearly any text network, unless heavy pre-processing is used, every document will have at least some similarity to every other document. Therefore, we can simplify this by dividing each element by 1 less than the N of documents in the corpus, k. This gives us the degree centrality.

However, because text networks are weighted, we use a slightly altered version of degree centrality suited for weighted networks which includes a tuning parameter α determining the relative importance of the number of total ties k compared to the weights of those ties a_{iq}.

The next step to get the proportional similarities is dividing the similarity between $Document_i$ and $Document_j$ by $Document_i$'s weighted degree centrality, x_i:

```
# set alpha to 1, the default
alpha <- 1
# calculate Opsahl et al's weighted degree centrality
k    <- rep(nrow(mat_a)-1, ncol(mat_a) )
a_iq <- colSums(mat_a)
x_i <- k * ( (a_iq/k )^alpha )
# divide ij by x_i
p_ij <- mat_a / x_i
```



As it relates to text networks, this can be interpreted as the extent that $Document_i$'s overall similarity to the corpus is explained by its similarity to $Document_j$. Therefore, most of $Document_1$'s overall similarity to the corpus is explained by it's similarity to $Document_2$.

Redundancy

The measure of redundancy is: $\frac{p_{qj}}{p_{iq}}$, where the proportional similarity of any third document q with j is divided by the proportional similarity of i to q. If the similarity between two neighbors of document i is high, this will penalize document i's textual spanning score in the final measure (and vice versa). Again, in a text networks, unless there is heavy pre-processing, all documents will have at least some similarity to all other documents.

If we want to know how redundant $Document_1$ is to $Document_4$, we can divide $Document_4$'s proportional similarity to all other documents (other than $Document_1$) by $Document_1$'s proportional similarity to all other documents (other than $Document_4$), and take the sum. In the long form, this is how we would get this:

```
# the Q in relation to J (Document 1) and I (Document 4) are Document's 2 and 3
q2   <- p_ij[2,1] / p_ij[4,2]
q3   <- p_ij[3,1] / p_ij[4,3]

i4_j1 <- q2 + q3
```

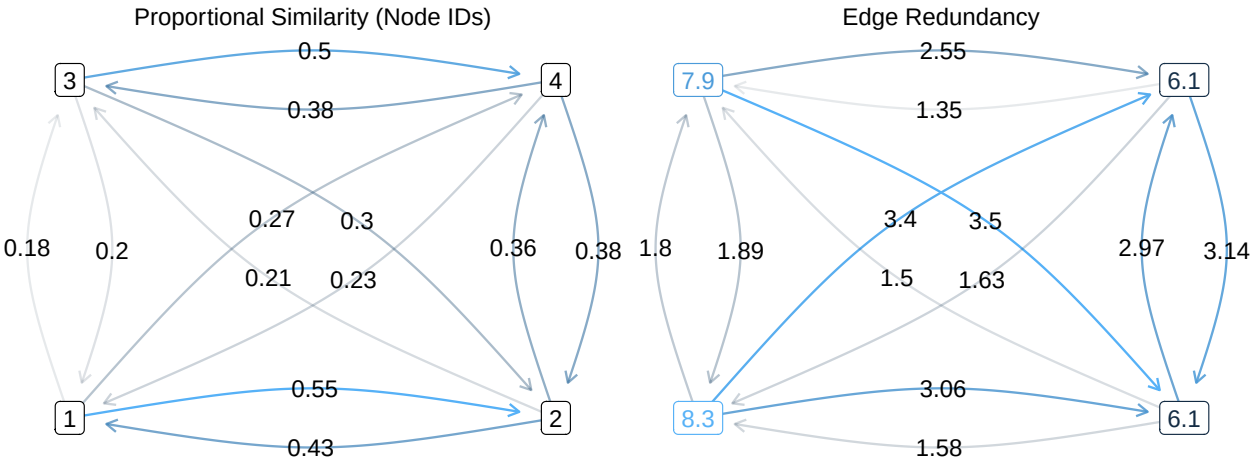
Redundancy
1.634286

We could do this one at a time, but there's a more efficient way. We know from network analysis that taking the second power of an adjacency matrix will give us the product of walks of length two (e.g., Newman 2018:131-2). However, we do not want to multiply the adjacency matrix by itself (`p_ij %*% p_ij`), we actually want to divide it by itself – but you can't just divide a matrix by itself. To accomplish this, we first take our adjacency matrix to the negative one power. Then we use matrix multiplication:

```
# find inverse:
ip_ij <- p_ij^-1
diag(ip_ij) <- 0
# matrix multiplication of adj. matrix and inv. matrix
r_ij <-  ip_ij %*% p_ij
diag(r_ij) <- 0
```

	1	2	3	4
1	0.000000	3.060256	1.803114	3.404762
2	1.579487	0.000000	1.501165	2.969697
3	1.890110	3.496504	0.000000	2.554113
4	1.634286	3.143636	1.345022	0.000000

The resulting edge-wise redundancies are in the table above. Note that the redundancy of $Document_1$ for $Document_4$ in the table is exactly the same as in the long form. We can then compare our proportional similarity measures to our edge redundancy (note vertex labels are the sum of all out-edges):



Dyadic and Cumulative Spanning

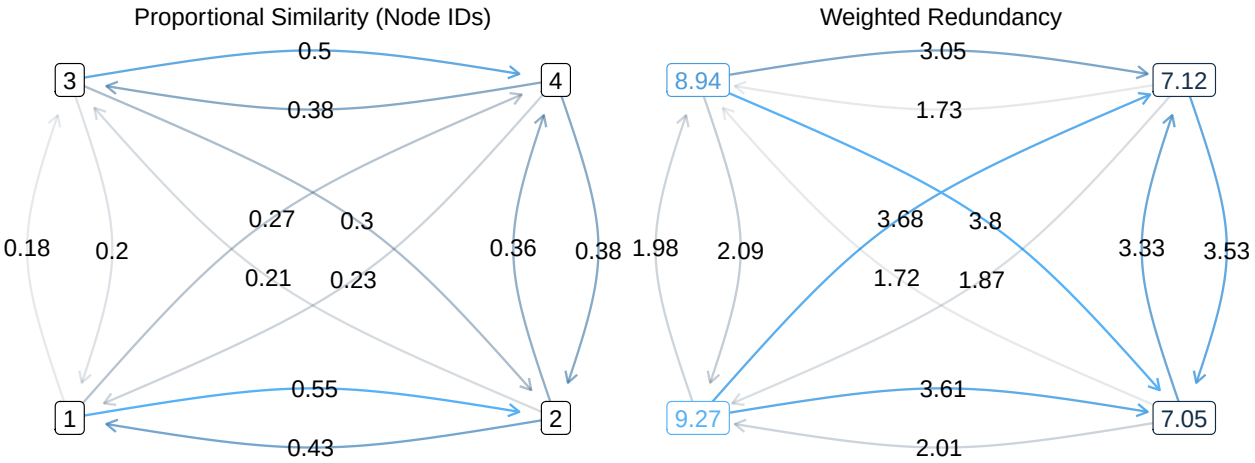
After we have a measure of the overall "redundancy" r_{ij}, we then add the proportional similarity p_{ij} to it's corresponding redundancy measure. This gives us a dyadic, or edge, spanning score d_{ij}. Which is straightfoward:

```
d_ij <- p_ij + r_ij
diag(d_ij) <- 0
```

	1	2	3	4
1	0.000000	3.605711	1.984932	3.677489
2	2.008059	0.000000	1.715451	3.326840
3	2.090110	3.796504	0.000000	3.054113
4	1.865055	3.528252	1.729637	0.000000

We are almost finished. We have weighted the proportional similarity of each edge by the corresponding redundancy of their out-edges, but we want a single score for each document summarizing how much it "spans" (or does not span) the text network. To get this, we first take the `rowSums` of the dyadic spanning matrix.

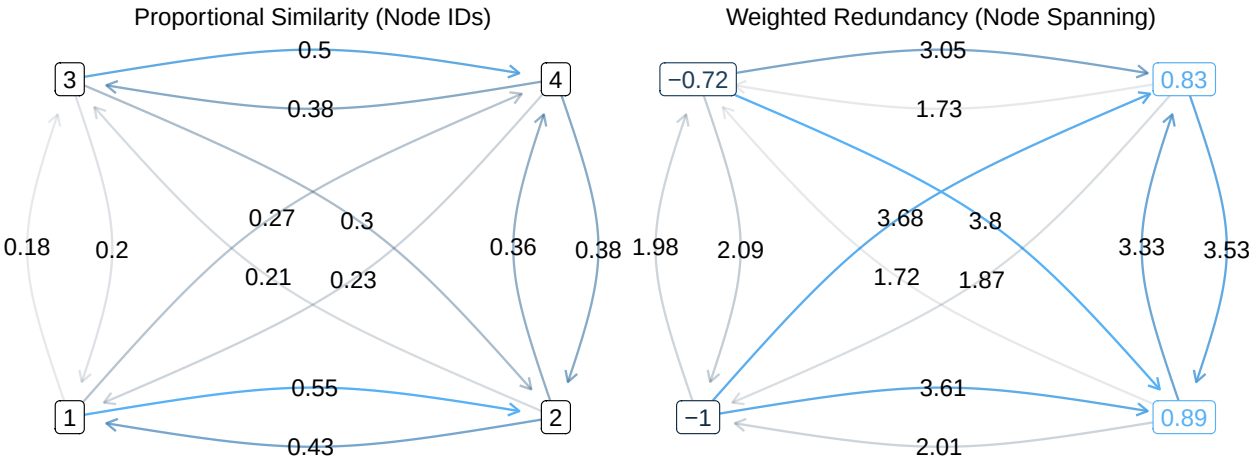
```
# cumulative spanning
cS_i <- rowSums(d_ij)
```



In the graph to the right, the nodes are labeled by their sum of their corresponding row in `d_ij` . We then standardize the scores. Finally, since higher numbers indicate greater redundancy, we reverse these scores to arrive at a measuring of overall spanning:

```
# normalize and invert the final score
cS_i <- scale(cS_i)[,1]*-1
```

Let's use the spanning scores to label the hypothetical documents in the graph. We expected $Document_2$ and $Document_4$ to be the high spanners, and our measure of textual spanning aligns with this expectation.



References

Burt, Ron. (1992). *Structural Holes: The Social Construction of Competition.* Cambridge, MA: Harvard University Press.

Opsahl, Tore, Filip Agneessens, and John Skvoretz. (2010). "Node Centrality in Weighted Networks: Generalizing Degree and Shortest Paths." *Social Networks.* 32(3):245–51

Stoltz, Dustin S., and Marshall A. Taylor. (2019). "Textual Spanning: Finding Discursive Holes in Text Networks." *Socius.* 5:2378023119827674.