1 Course Description

Network analysis is an increasingly important analytic approach utilized to find structure in relationships that, due to their complexity and/or dimensionality, are hidden at first glance. The goal of this course is to teach participants how to harness the mathematical power of network analysis in order to find structure in written content. This analytic method (Network Analysis of Qualitative Data — NAQD) perfectly and powerfully blends quantitative, mathematical, and qualitative principles to analyze text data, which is an approach yet to be broadly implemented in education research.

The course relies on both lecture and hands-on exercises. All data and software are freely available, which liberates participants from the need to spend hundreds or thousands of dollars to conduct analyses. Throughout the course, the analysis and interpretation of data will be prioritized. Once participants have mastered the network concepts covered — including measures of centrality and network visualization— the course will move toward the application of these principles and techniques to highlight the most important actors, codes, and words in a dataset.

This course is appropriate for graduate students, early career scholars, and advanced researchers. The course will be based on the use of R, but requires no previous familiarity with the software. A quick R tutorial will be provided along with all the code and data required to replicate the exercises. Participants are required to bring their computers and should be able to install software. Assignments entail replicating and interpreting the in-class analyses with an emphasis on the relevance of discovered networks.

2 Course Faculty

Instructor: Manuel S. Gonzalez Canche
Assistant Professor
Institute of Higher Education
University of Georgia
Meigs Hall 116
✉️: msgc@uga.edu, ☎️: (706) 583-0048

2.1 Course Format

Preferred format: Extended session, 16-hour session. Also, please consider including this course in the AERA- VRLC.

---

1A T.A. will be sponsored by the Institute of Higher Education, University of Georgia.
3 Proposal Narrative and Relevance

In early 2016, NVivo announced that one of its platforms would offer the possibility of conducting social network analysis. This capability, however, was constrained to the most expensive product (NVivo 11 Plus) which ranges in price from $1,900 to $950 for non-education-affiliated and education-affiliated users, respectively. Notably, these prices did not include extended support, nor a fundamental course about how to use the most recent network analysis features. These support services have a combined additional price of $710 and $355 per year for non-education-affiliated and education-affiliated researchers or analysts. It is also worth noting that NVivo offered a webinar to highlight the network analysis features in order to entice researchers to purchase NVivo 11 Plus as this is the only platform with network analysis capabilities. In spite of this product’s price tag, several features essential to the analysis and visualization of network data are currently missing from its capabilities. For example, this product cannot highlight network structure by changing the colors of actors, and it is incapable of adding size to actors in the network or the links between them. As a consequence, these new features of NVivo represent an empirically limited research tool.

This workshop is aligned with AERA’s meeting goal of putting “Knowledge to Action [and] Achieving the Promise of Equal Educational Opportunity.” All education researchers, regardless of means to pay for software, should be equipped with the tools to make decisions based on robust evidence. Accordingly, this course offers a cost-free option to conduct network analysis of qualitative data using the R platform. This workshop has successfully been implemented four times at the University of Georgia and based on its excellent reviews and evaluations was offered for the first time in June 2016 at UCLA’s Higher Education Research Institute (HERI) with a record attendance of 32 registered participants and 10 more graduate students and HERI staff.

3.1 Prerequisite Skills or Knowledge

This course is appropriate for graduate students, early career scholars, and advanced researchers. Both quantitative and qualitative researchers are welcomed to participate. Ideally, all participants should be familiar with coding qualitative data retrieved from interviews, observations, or essays. The seminar can accommodate up to 35 participants.

3.2 Rationale

This course aims to provide researchers with quantitative tools to analyze text or written data gathered from a variety of sources (e.g., newspapers, journals, books, tweets, memos, interviews, or annual reports). Given the abundance of texts and other types of data that can be coded, and considering that network analysis was designed to extract seemingly absent patterns across relationships, the use of network principles to study actors, codes, and texts, has the potential to enhance the analytic research process by supporting or corroborating the results obtained through qualitative analytic techniques. Specifically, after applying network techniques to written information, researchers will be able to observe the structure of qualitative data and use this output as an empirical guide to analysis based on mathematical principles. NAQD is not meant to replace the researcher’s expertise gained
prior to or during data collection. On the contrary, the understanding obtained from the analysis of text network outputs is enhanced and complemented with theoretical lenses, previous knowledge/literature, and additional data collected by the researcher. Another reason for the salience of the proposed course stems from its reliance on free software and tools, which liberates participants from the need to spend considerable amounts of limited research funding to conduct network analyses of qualitative data.

3.3 Learning Objectives

At the end of the course, all participants are expected to have developed the skills and knowledge necessary to conduct NAQD in order to highlight the structural relationships among participants, their qualitative codes, and the text configuring each code. Figure 5, in the supporting materials section reflects the most comprehensive expected final product.

The successful attainment of the learning objectives requires the development of specific skills and knowledge. The following section describes the plan to achieve the stated goal of the course along with the time that each step will require. The time and plan are based on previous implementations of the course with doctoral students from both different academic backgrounds (teaching education, psychology, computer science, mathematics education, public policy, political science, and higher education) and levels of proficiency in R. The topics covered are part of the class titled “Statistical Network Analysis” that has successfully been implemented at the University of Georgia during Summer of 2013, 2014, 2015, and 2016 (please see a summary of course evaluations in the supporting materials section).

3.4 Course Content

In the first section of the course, participants will be introduced to the R programming language environment. During this section, participants will install R and R for Qualitative Data Analysis (RQDA) freeware. Once the installation is complete (30 minutes), selected sections of the R tutorial attached to the supporting materials section will be discussed (2 hours). The goal of the tutorial is for participants to become comfortable reading the R programming code to be used during the remainder of the course.

The next section of the course is conceptual (3 hours). The goal of this section is for participants to understand the notions of centrality, types of networks, community detection algorithms, and the importance of all of these topics in highlighting structural relationships. Three main measures of centrality will be discussed: degree, betweenness, and eigenvector centrality. These measures were selected given that they convey different pieces of information and thus highlight different parts of network structure across relationships. It is particularly important that participants demonstrate an understanding of how these centrality measures will vary based on the types of relationships being analyzed (e.g., person-to-person or person-to-organization, i.e., one-mode and two-mode networks, respectively). Finally, this section will also cover community detection algorithms (e.g., Cluster Optimal and Fast and Greedy) with emphasis on the rationale followed in each, which, once again, will enable researchers to emphasize different attributes of the relational information under study.

All of these concepts will be combined in the third course section using a technique called Key Actor Analysis (2.5 hours). After a brief presentation of the rationale guiding
this analytic approach, participants will apply this technique to real data. The goal is that all participants will successfully replicate these procedures (shown in Figures 1 and 2) and write a one-paragraph report on their findings (the supporting materials section contains examples of handouts on community detection and key actor analysis).

The next topic will cover Network Analysis for Qualitative Data, which includes the coding of written information using qualitative principles (4 hours). As shown in the supporting materials section, coded texts contain rich contextual information that is translated into a set of relationships (participant→code) that can be analyzed following two-mode or affiliation network principles. The goal of this exercise is for all participants to practice coding five to ten one-page essays about students’ reasons for leaving school. These essays were collected as part of the School Leavers Study, 1978 (Pahl, R.E., School Leavers Study, 1978. 2nd Edition. Colchester, Essex: UK Data Archive, June 2012. SN: 4867, doi:10.5255/UKDA-SN-4867-1) and are publicly available. After these essays have been coded, their relational form will be visualized graphically (a sociogram) following the principles and procedures conducted in the Key Actor Analysis session and as shown in Figure 3.

The last topic will cover text-mining techniques employed to extract the most important words associated with each code (4 hours). The mining procedures will enhance the visual representation of codes and their texts as shown in Figure 4. Once again, all data and coding schemes required to conduct the analyses will be provided. Participants will also be required to write a one-paragraph summary of Figure 4. Particular emphasis will be placed on asking participants to highlight the structure that emerges from applying community detection algorithms to the word frequencies constituting each code. The final deliverable accounts for a three-mode network in which the elements are participants, codes, and words configuring each code. All of these elements are then represented in a single sociogram. One of the most important components of the final product of the course is shown in Figure 5, where triangles account for codes, circles represent the most important words, and squares account for individuals affiliated to both codes and words. In this sociogram, for example, one can immediately identify the codes “Utility and Applicability” and “Skill-set Building” as the most important reasons the participants gave for participating in the most recent Statistical Network Analysis class offered in 2016 at the University of Georgia. The figure also shows which participants were closer to those reasons and what pieces of information were peripheral. A completely interactive version of Figure 5 can be found in the following hyper-link: http://ihe.uga.edu/networkD3/. The remaining parts of this final deliverable will include a one-paragraph interpretation of the sociogram shown in Figure 5 and another where participants discuss the ways in which they can utilize this technique in their own research and how this method can enhance their traditional qualitative analysis practice.

3.5 Class Dynamics

Each topic will start with a discussion of the method to be addressed. Due to the intensive nature of the course, the mathematical underpinnings of the methods will not be covered in-depth. Rather, discussion will be focused on conceptual understandings of the purposes of the models, the contexts in which they can be applied, and the interpretation of output. The second part of each topical session will be devoted to replicating the analyses discussed in the first half with paragraph deliverables requested at the end of these replication exercises.
4 Course Supporting Materials

All materials, including presentations, R code, data, and software will be provided. The software is available for Mac and PC users at absolutely no cost.

4.1 Recommended Books, Articles, and Presentations

Prior to workshop attendance, participants should read the references in bold font. The remaining materials are optional yet highly valued in Network Analysis.


Other recommended references

Course Agenda and Lessons Plan
# Network Analysis of Qualitative Data

<table>
<thead>
<tr>
<th>Lecture Number</th>
<th>Lecture Name</th>
<th>Aims and Objectives</th>
<th>Activities, Deliverables</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Topic I: Freeware Installation and Tutorial (2.5 hours)</strong></td>
<td></td>
<td>1. Download and install R and RQDA</td>
</tr>
</tbody>
</table>
|                | 1. Installation of R, RQDA and R Tutorial | Aim: To install all freeware required for the successful replication of analyses and to familiarize participants with R commands to be used.  
Objectives: By the end of this lecture a student should be able to:  
1. Access R and RQDA in their own computers  
2. Understand the fundamental structure of R  
3. Read data into R, and  
4. Understand the functions of the commands explained in class. | 2. Download R tutorial  
3. Note taking about command explanation                                                                          |
|                | **Topic II: Introduction to Network Analysis Concepts (3 hours)** | Aim: To introduce the notions of centrality, types of networks, and community detection algorithms.  
Objectives: By the end of this lecture a student should be able to:  
1. Understand the differences between degree, betweenness, and eigenvector centrality measures and the different types of information they convey  
2. Understand the differences between one mode, two-mode, and three-mode networks with emphasis on how the measures of centrality would vary given the network structure  
3. Understand the conceptual differences among the three community detection algorithms employed — Fast and Greedy, Edge Betweenness, Random Walk— emphasizing the information they convey. | 1. Open handouts prepared for the session  
2. Note taking about explanation  
3. Ask questions and/or contribute to the discussion                                                                 |
<table>
<thead>
<tr>
<th>Lecture Number</th>
<th>Lecture Name</th>
<th>Aims and Objectives</th>
<th>Activities, Deliverables</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Topic III: Key Actor Analysis (2.5 hours)</td>
<td><strong>Key Actor Analysis</strong>&lt;br&gt;Aim: To introduce participants to the notion of key or central actor analysis, emphasizing its relevance for the Network Analysis of Qualitative Data.&lt;br&gt;Objectives: By the end of this lecture a student should be able to:&lt;br&gt;1. Understand the relevance of measures of centrality, types of networks, and community detection in the identification of central players&lt;br&gt;2. Understand the relevance of key actor analysis in the Network Analysis of Qualitative Data&lt;br&gt;3. Understand the functions of the commands explained in class&lt;br&gt;4. Replicate the analyses conducted during the session.</td>
<td>1. Replicate Figure 1&lt;br&gt;2. Write a one-paragraph analysis of Figure 1&lt;br&gt;3. Replicate Figure 2&lt;br&gt;4. Write a one-paragraph analysis of Figure 2</td>
</tr>
<tr>
<td></td>
<td>Topic IV: Network Analysis of Qualitative Data I (4 hours)</td>
<td><strong>Introduction to qualitative data coding in R and Network Analysis of Qualitative Data</strong>&lt;br&gt;Aim: To introduce students to coding qualitative data in R and detecting key codes and actors within datasets&lt;br&gt;Objectives: By the end of this lecture a student should be able to:&lt;br&gt;1. Read qualitative data in R&lt;br&gt;2. Create a new project using R for Qualitative Data Analysis (RQDA)&lt;br&gt;3. Code five to ten one-page essays about reasons to leave school&lt;br&gt;4. Understand the resulting relationship structure between codes and participants (two-mode network)&lt;br&gt;5. Understand the importance of key actor analysis in the Network Analysis of Qualitative Data&lt;br&gt;6. Replicate the Network Analysis procedure shown in Figure 3</td>
<td>1. Read qualitative data in R&lt;br&gt;2. Replicate Figure 3&lt;br&gt;3. Write a one-paragraph analysis of Figure 3</td>
</tr>
<tr>
<td>Lecture Number</td>
<td>Lecture Name</td>
<td>Aims and Objectives</td>
<td>Activities, Deliverables</td>
</tr>
<tr>
<td>----------------</td>
<td>--------------</td>
<td>---------------------</td>
<td>--------------------------</td>
</tr>
</tbody>
</table>
| 5.             | Text-Mining Techniques and Network Analysis of Qualitative Data | Aim: To introduce the notions of text-mining techniques of codes and add these results to the Network Analysis of Qualitative Data. Objectives: By the end of this lecture a student should be able to:  
1. Apply text-mining techniques to identify the most important words of each of the codes identified in Figure 3  
2. Understand that codes and the words configuring each code naturally converge into a two-mode network, wherein words are affiliated to codes.  
3. Replicate Figure 4 highlighting its usefulness and limitations.  
4. Form a three mode network configured by Actors, Codes, and Words.  
5. Replicate Figure 5, which is based on key actor analysis  
6. Discuss cases in which the application of these analytic techniques is appropriate for her/his own research agenda | 1. Text-mine coded texts  
2. Replicate Figure 4  
3. Write a one-paragraph analysis of Figure 4  
4. Replicate Figure 5  
5. Write a one-paragraph analysis of Figure 5  
6. Write a paragraph on other settings where the skills developed as a result of the course will be appropriate |
One-page Relevant CV
Employment

Assistant Professor of Higher Education (Since August, 2012).
The Institute of Higher Education, The University of Georgia

Education

Certified reviewer What Works Clearinghouse (November, 2014)
Mathematica Policy Research/Institute of Education Sciences

Ph.D. Higher Education, cognates in Biostatistics and Economics (June, 2012)

Research Interests

- Community colleges
- Underrepresented students’ educational and occupational trajectories
- Higher Education finance and policy
- Quantitative research methods (quasi-experimental, spatial, and network analysis)

Research Relevant to Network Analysis


Teaching, Doctoral level

Transitions into and out of college. Spring term: 2013

Workshops in Advanced Methods in Network Analysis.
June 25-27, 2012: University of Kassel, Germany. Participants came from Portugal, Croatia, Germany, Italy and Finland.
February 12, 2016: University of Georgia, U.S.A. Workshop Offered to Graduate Students attending the Graduate Student Association’s Interdisciplinary Research Conference.
June 28-29, 2016: University of California Los Angeles, U.S.A. Participants came from across the United States.
Course Evaluations
Table 1. Summary of Student Evaluations for Doctoral Courses, Fall 2012 – Maymester 2016 (scale ranges from 0 to 5 points)

<table>
<thead>
<tr>
<th>Courses</th>
<th>Quant I (F12, F13, F14, F15)</th>
<th>Quant I Lab (F12, F13, F14, F15)</th>
<th>Network Analysis (M13, M14, M15, M16)</th>
<th>Quasi-experimental (S13, S14, S16)</th>
<th>Transitions into and out of college (S13)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Course met its learning objectives</td>
<td>4.9</td>
<td>4.7</td>
<td>4.8</td>
<td>4.6</td>
<td>4.9</td>
</tr>
<tr>
<td>Instructor provided quality feedback</td>
<td>4.8</td>
<td>4.9</td>
<td>4.9</td>
<td>4.6</td>
<td>4.6</td>
</tr>
<tr>
<td>Instructor provided timely feedback</td>
<td>4.7</td>
<td>4.9</td>
<td>4.9</td>
<td>4.6</td>
<td>4.2</td>
</tr>
<tr>
<td>Instructor demonstrated subject expertise</td>
<td>5.0</td>
<td>4.8</td>
<td>4.9</td>
<td>5.0</td>
<td>4.9</td>
</tr>
<tr>
<td>Course assignments and activities helped me learn</td>
<td>4.7</td>
<td>4.5</td>
<td>4.9</td>
<td>4.6</td>
<td>4.6</td>
</tr>
<tr>
<td>Course challenged me to think</td>
<td>4.9</td>
<td>4.8</td>
<td>4.8</td>
<td>4.7</td>
<td>5.0</td>
</tr>
<tr>
<td>n</td>
<td>54*</td>
<td>38</td>
<td>60</td>
<td>24</td>
<td>12</td>
</tr>
</tbody>
</table>

F = Fall, M = Maymester, S = Spring, *15 EDD students in summer 2014

In addition to the numeric rating, student comments on anonymous end-of-course evaluations provide additional information about the effectiveness of these courses. Relevant to the proposed course, what follows are statement provided by students enrolled in Statistical Network Analysis.

“Dr. Canche’s expertise and knowledge of the subject matter were most valuable.”
“Dr. Gonzalez Canche has the gifts of patience and grace under pressure. His ability to quickly transfer knowledge of difficult/challenging material should be applauded.”
“Perhaps most valuable was the countless hours that Dr. Canche must have spent developing the exercises for the portfolio and supporting students’ efforts to complete the components of the portfolio, he was extremely accessible and dedicated to us as students.”
“An interesting topic, and relevant course content, acquiring new methodological skills, very effective teaching skills of the instructor, comfortable and supporting learning environment.”
“This course was a great methods class and introduction to R, I liked how we systematically went through the process of learning R and the exercises we were expected to complete.”
“I truly appreciate the professor’s knowledge and expertise in what is a burgeoning field.”
Examples of Previous Materials and Main Presentation
Background

- Ph.D. in Higher Education with cognates in Biostatistics and Economics
- In biostats I took classes on analysis of high dimensional data analysis which consists of mining quantitative data
- Six years ago I transitioned to start mining text data which is the origin of this approach
- Text mining allows us to find structure of the data,
- However, it is limited (e.g., word frequencies, cluster, correlations, dendograms).
- More importantly, text mining per se loses the context in which the text was mentioned
- Example:
  - Analysis of 198 presidential speeches over 31 years of data.
  - Purpose was to mine the data to find the structure and evaluate any potential change.

Examples
Rationale of Network Analysis of Qualitative Data

Figure 1: Logic model guiding the approach, implies that we need to talk about qualitative coding, network principles, and their integration.

Notes

Conceptualization of Network Analysis

- **Network Analysis:** the understanding of links or connections among units relying on mathematical principles

- **Units can be of any type or form**

- We can analyze the connections of
  - Students to students (friendship, classrooms, romantic relationships, fights . . .).
  - Universities to universities (based on distance, ascription to tuition reduction agreements, NCAA).
  - Combinations, e.g., students attending universities . . .

- **Take home point** is that this method allows one to think outside the box and identify innovative sets of relationships that can be analyzed mathematically

- Today we will focus on analyzing qualitative data: texts, interview, observation, . . .
Background
Session
Structure
Network
Analysis
Traditional Network Structure
Centrality Measures
More traditional, yet novel SNA approach
Integration with qualitative data
Qualitative Network Structure

Notes

Examples from a recent project...

Net of reasons to enroll in a CC

Types of Data Structure

Institutional array of college classes (edge list)

<table>
<thead>
<tr>
<th>ID</th>
<th>ClassID</th>
<th>SemID</th>
<th>ID</th>
<th>ClassID + SemID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>xrr</td>
<td>113</td>
<td>1</td>
<td>xrr_113</td>
</tr>
<tr>
<td>1</td>
<td>xrt</td>
<td>121</td>
<td>1</td>
<td>xrt_121</td>
</tr>
<tr>
<td>1</td>
<td>vrt</td>
<td>113</td>
<td>1</td>
<td>vrt_113</td>
</tr>
<tr>
<td>2</td>
<td>xrr</td>
<td>113</td>
<td>2</td>
<td>xrr_113</td>
</tr>
<tr>
<td>2</td>
<td>erv</td>
<td>121</td>
<td>2</td>
<td>erv_121</td>
</tr>
<tr>
<td>2</td>
<td>web</td>
<td>121</td>
<td>2</td>
<td>web_121</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Matrix representation of the edge list

\[
\begin{pmatrix}
N & N & N & N & N \\
1 & 2 & 0 & 0 & 0 \\
2 & 2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
\end{pmatrix}
\]

Highlighting Network Structure

- Given the dynamic nature of social groups the identification of key players is critical in any attempt to influence the behavior of the network.
- Key actor analysis has been used to identify influential or peripheral actors that if removed would either dismantle or produce the least change in the network, respectively (more on this later).
Quick Overview of Centrality Measures

The identification of key actors relies on:

**Degree**: Number of connections an actor has (Facebook friends)

**Betweenness**: Proportion of times an actor lies in between two other actors. This makes this actor important in controlling the flow of information in the network. A.K.A as gatekeeper (to reach someone, you have to go through this actor)

**Eigenvector**: A measure of how central an actor is and how central the ties of this actors are in the network. This unit is central because she/he is friends with other really central units.

Methodologically, all these measures come from the same matrices and to some extent share mathematical properties, but diverge from linear relationships.

Adding These Measures as Attributes

Figure 2: As mentioned earlier, the default option is not revealing much!

Using centrality measures we can identify key actors

- An actor with very high betweenness but low EC may be a critical gatekeeper to a central actor
- Likewise, an actor with low betweenness but high EC may have unique access to central actors

A Better Approach to Highlight Structure

Figure 3: Key actors weighted betweenness centrality, name reveals top 10% of eigenvector centrality
This approach works regardless of complexity

Structure of Qualitative Data

- A.K.A as unstructured data
- Algorithms that capture context are still in development
- Qualitatively coded data does possess the structure required to implement network principles
- Qualitative data is rich in contextual information.
- Consequently, by analyzing these data using network techniques, we can mine the structure of these relationships.
  - Participant and Codes
  - Codes and the text configuring each code
  - Participants who shared a given code
  - Relationships across codes
- All these procedures (even qualitative coding) can be executed with free software.

Types of Data Structure

**Edge list** or list of codes and content

<table>
<thead>
<tr>
<th>ID</th>
<th>Code</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Career</td>
<td>At various times . . .</td>
</tr>
<tr>
<td>1</td>
<td>Utility_Applability</td>
<td>Practically, this . . .</td>
</tr>
<tr>
<td>2</td>
<td>Utility_Applability</td>
<td>I want to learn . . .</td>
</tr>
<tr>
<td>2</td>
<td>Data_Visualization</td>
<td>I expect to learn . . .</td>
</tr>
<tr>
<td>3</td>
<td>Utility_Applability</td>
<td>become more comfortable . . .</td>
</tr>
<tr>
<td>3</td>
<td>Skill_Set_Building</td>
<td>develop a better . . .</td>
</tr>
<tr>
<td>3</td>
<td>Utility_Appliability</td>
<td>uses of these models . . .</td>
</tr>
<tr>
<td></td>
<td>Utility_Appliability</td>
<td>this course will aid . . .</td>
</tr>
</tbody>
</table>

Matrix representation of the edge list

<table>
<thead>
<tr>
<th>ID</th>
<th>Career</th>
<th>Utility_Applability</th>
<th>Data_Visualization</th>
<th>Skill_Set_Building</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>N</td>
</tr>
<tr>
<td>N</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>N</td>
</tr>
</tbody>
</table>
Example 1 Reasons to Take SNA

Data Visualization

Computer Infrastructure

Research Presentation

Network Analysis

Size conditional on EV, color indicates between centrality

Figure 9. QNA, Actors and Codes

Example 2 Reasons to Leave School

Figure 4: Analysis conducted by Michael Snell, SNA, 2015

Example 2.1 Reasons to Leave School

Notes
When we combine our actors, their words, and the codes we created, there emerges a recurring theme across most of the essays we used as input for our qualitative network analysis:

In most cases, leaving school was closely connected to language related to jobs. We see the term "apprenticeship" and even the more particular phrase "got apprenticeship" over and over. We also find that financial concerns are often in close proximity, and these central motivating factors in the decision of whether or not to leave school.

These results correspond well with other information that exists independent of our essays. During the timeframe during which most students left school, there was a severe shortage of jobs, with apprenticeships becoming a major step in securing a future position at a given place of work.

These results correspond well with other information that exists independent of our essays. During the timeframe during which most students left school, there was a severe shortage of jobs, with apprenticeships becoming a major step in securing a future position at a given place of work.
Hierarchical Clustering: Community Detection

Manuel S. González Canché
msgc@uga.edu

Most methods for graph partitioning are essentially variations on the more general concept of hierarchical clustering. These methods take a greedy approach to searching the space of all possible partitions. A greedy algorithm is an algorithm that follows the problem-solving heuristic of making the locally optimal choice at each stage with the hope of finding a global optimum. Greedy usually yields locally optimal solutions that approximate a globally optimal solution in a reasonable time.

Hierarchical methods are classified as either agglomerative or divisive. Agglomerative forms partitions by merging elements locally. Divisive is based on the successive refinement of partitions through the process of splitting. Commonality: at each stage, the current candidate partition is modified in a way that minimizes a specified measure of cost. In agglomerative methods, the least costly merge of two previously existing partition elements is executed. In divisive methods, it is the least costly split of a single existing partition element into two that is executed.
More formally...

- One of the most important measures is modularity.
- Networks with high modularity have dense connections between the nodes within modules but sparse connections between nodes in different modules.
- Let $P = \{C_1, \ldots, C_k\}$ be a given candidate partition and define $f_{ij} = f_{ij}(P)$ to be the fraction of edges in the original network that connect vertices in $C_i$ with vertices in $C_j$.

$$\text{mod}(P) = \frac{1}{2m} \sum_{k=1}^{k} \left[ f_{kk}(P) - f_{kk}^* \right]^2,$$

where $f_{kk}^*$ is the expected value of $f_{kk}$ under some model of random edge assignment.

More on modularity

- Large values of the modularity are therefore taken to suggest that $P$ captures nontrivial ‘group’ structure, beyond that expected to occur under the random assignment of edges.
- In principle the optimization of the modularity requires a search over all possible partitions $P$, which is prohibitively expensive in networks of moderate size and larger.
- Modularity is often used in optimization methods for detecting community structure in networks. However, it has been shown that modularity suffers a resolution limit and, therefore, it is unable to detect small communities.
- igraph conducts the modularity in the fastgreedy.community algorithm to produce an object of the class communities, which can then serve as input to various other functions.

Fast and greedy...

Figure 1: There are two factions, yet three communities.
**Dendogram representation...**

![Dendogram](image)

*Figure 2: More traditional approach*

---

**Edge betweenness approach**

- The edge betweenness score of an edge measures the number of shortest paths through it.
- It is likely that edges connecting separate modules have high edge betweenness as all the shortest paths from one module to another must traverse through them (bridges).
- We can use the information provided by those bridges to detect communities using modularity principles.
- `edge.betweenness.community` performs this algorithm by calculating the edge betweenness of the graph, removing the edge with the highest edge betweenness score, then recalculating edge betweenness of the edges and again removing the one with the highest score, etc.

---

**Edge betweenness community...**

![Edge betweenness community](image)

*Figure 3: There are two factions, yet x number of communities*
Random walk community

- Given an undirected graph $G = (V, E)$ and some starting vertex $s$, a random walk in $G$ of length $t$ is defined as a randomized process.
- Starting from the vertex $s$, we repeat $t$ times a step that consists of choosing at random one of the neighbors $v'$ of the vertex $v$ we are at and moving to it.
- The goal consists of estimating the probability that we will end up in that randomly chosen vertex $v'$ in $t$ number of steps.
- This definition is also valid for a directed graph.
- Random walks as defined above are naturally viewed as trajectories determined by the outcome of randomized choices we make at each step.

More formally

- Probability distributions are calculated about mobility among neighbors.
- For a given vertex $v$, we define $p_{v'}^t$ to be the probability that we are at vertex $v'$ after $t$ steps of the random walk.
- The walktrap community finding algorithm is based upon this notion.
- This function tries to find densely connected subgraphs, also called communities in a graph via random walks.
- The idea is that (probabilistically and empirically) short random walks tend to stay in the same community.

Walktrap community...

![Figure 4: There are two factions, four communities]
Optimal Solutions

- Spin glass community and cluster optimal render robust results, similar to the random walk solution. Of these three approaches cluster optimal is by far the most computationally expensive.
- Cluster_optimal maximizes the modularity measure over all possible partitions. The modularity maximization is a programming problem and external libraries are called to solve for it. In graphs up to 50 vertices cluster_optimal should work fine.
- The spin glass community uses a spin glass model and simulated annealing.
- This function allows for partitioning the vertices into communities, while optimizing an energy function.
- The idea is the same as in the random walk approach set of nodes with many edges tending to form a community.

Notes

(a) Spin glass community

(b) Cluster optimal community

Notes
Purpose and Outline

- SNA is often used to identify central or key actors within a social group.
- Given the dynamic nature of social groups, the identification of key players may be critical in any attempt to influence the behavior of the network.

**Purpose:** At the end of the session, the participants are expected to understand and replicate the procedures followed to conduct key actor analysis in R.

We will use two centrality measures and linear regression procedures to conduct the analysis.

Quick review of centrality measures

The identification of key actors relies on:

- **Degree:** Number of connections an actor has
- **Betweenness:** Number of shortest paths an actor is on, which makes this actor important in controlling the flow of information in the network.
- **Closeness:** Relative distance of one actor to all other actors.
- **Eigenvector:** A measure of how central an actor is and how central the ties of this actor are in the network.

- **Methodologically,** all these measures come from the same matrices and to some extent share **mathematical properties**.
- Consequently, these measures are expected to have close to linear relationships.
Building upon linear relationships of centrality measures

Plotting centrality measures...

- A method for using centrality metrics to identify key actors is to plot actors' scores for Eigenvector centrality (EC) versus Betweenness.
- Although these measures are correlated, they are not perfectly linear.
- In this sense, we can use these non-linear outliers to enrich our knowledge of the social relationships in the network.

**Conceptual implications**

- An actor with very high betweenness but low EC may be a critical gatekeeper to a central actor.
- Likewise, an actor with low betweenness but high EC may have unique access to central actors.

![Diagram](image_url)

**Figure 1:** Key Actor analysis, this is part of what we will replicate today!

---

**Adding these measures as attributes in a sociogram**

**Using centrality measures to highlight key actors**

- Using our network data we will identify the location of the key actors from the previous analysis.
- Theoretically, our interpretation should be completely congruent with what was found in the bivariate plot.
- We can use the regression residuals if we want to, or simply use eigenvector or betweenness centralities as attributes, conditional on what we want the sociogram to show.

![Sociogram](image_url)

**Figure 2:** As usual, the default option is not revealing much!
Review of degree measures
Integrating statistics to enrich SNA
More traditional, yet novel SNA approach

Figure 3: Key actors weighted by eigenvector and betweenness centrality

Thank you!
Contact information
msgc@uga.edu
Detailed examples of activities and/or exercises
Figure 1: Key Actor Analysis: Actors weighted by eigenvector and betweenness centrality
Figure 2: Scatter plot representation of Key Actor Analysis
Figure 3: Qualitative Network Analysis (two-mode network). The elements of this network are codes (circles) and participants (squares). Size conditional on eigenVector centrality, color indicates betweenness centrality. All names are pseudonyms and text data were retrieved from participants’ stated motives for studying Statistical Network Analysis in Summer 2016.
Figure 4: Qualitative Network Analysis (two-mode network). The elements of this network are codes (circles) and the words (squares) configuring each code. Size conditional on eigenvector centrality, color indicates betweenness centrality. Text data were retrieved from participants’ stated motives for studying Statistical Network Analysis in Summer 2016.
Figure 5: Qualitative Network Analysis (three-mode network). The elements of this network are participants (squares), codes (triangles), and the words (circles) associated with each code. Size conditional on eigenvector centrality, color indicates betweenness centrality. All names are pseudonyms and text data were retrieved from participants’ stated motives for studying Statistical Network Analysis in Summer 2016. A completely interactive version of this figure can be found in the following location: http://ihe.uga.edu/networkD3/.
Figure 6: Qualitative Network Analysis (three-mode network). The elements of this network are the same as those contained in Figure 5. Size conditional on eigenvector centrality, color indicates membership resulting from the community detection algorithm employed (Cluster Optimal). This representation is particularly useful when the goal is to identify participants’ ascription to communities based on word and code frequencies. Please follow this link (http://ihe.uga.edu/networkD3/) for an interactive version of this sociogram. Participants will be provided with the code to obtain this interactive sociogram.
1 Fundamentals and Installation Of R

1.1 Installation .................................................. 2
1.2 RStudio installation .......................................... 3
1.3 notepad++ ....................................................... 3
1.4 R & RStudio .................................................... 4

2 a <- “hello world”

2.1 Getting help ................................................... 5
2.2 Work-flow ....................................................... 5
2.3 Number and Vectors .......................................... 6
2.4 Some commonly used functions ............................ 8
2.5 Matrices, Lists and Data Frames ......................... 12
2.6 Reading external files ....................................... 19
2.7 Exploring data frames ...................................... 19
2.8 Data Analysis .................................................. 20
2.9 Some basic inferential stats ................................. 22
2.10 Programming in R (Chapters 9 and 10) .................. 25
2.11 Packages (Chapter 13) ...................................... 26

*The lecture notes are based on “An Introduction to R” by W. N. Venables, D. M. Smith, and the R Development Core Team that can be accessed here. You can find more details in that on-line book. Other great resources are the “R Reference Card V1” by Tom Short that can be found here and the “R Reference Card V2” by Matt Baggott that can be found here.
Why to use R

R is very flexible and powerful tool for statistical analysis. It is both a programming language and a statistical platform. It gives flexibility that allows users to analyze their data more creatively and sometimes in a more advanced way.

This manual is by no means comprehensive, it merely introduces participants to have a working understanding of the R language. It will allow us to comfortably read and manipulate R code to adapt it to their own projects.

Note: This document is only for reference, it is not recommended to copy any commands from here. The code contained is in R language and requires “perfection.” Copying and pasting from this document is likely to contain noise and consequently the functions won’t work, or won’t work properly. Instead create a text document called “R functions Manual.R” to be used during the session which will be described later in this tutorial.

1 Fundamentals and Installation Of R

Even though R could be difficult to learn . . .

- Has THE MOST advanced techniques in many disciplines, including network analysis and quasi-experimental design.
- It is flexible, powerful, and free

Shortest R history

- R is an open source language and environment for statistical computing and graphics.
- Ross Ihaka and Robert Gentleman (University of Auckland, New Zealand) did not want their students to pay to analyze data and do graphs.
- Given that their both names started with R, they name the new project R, in part to make fun of S.
- R, like S, is designed around a true computer language, allowing users to continue creating functions and packages.
- R has more than 5,000 packages... and continues growing.

1.1 Installation

(a) Open your web browser and go to http://www.r-project.org/

(b) On the left panel, click “CRAN” under “Download” and choose a mirror site close to us from the list, say http://rweb.quant.ku.edu/cran/

(c) Under “Download and Install R”, choose your operating system,

(d) Windows users can go here http://cran.rstudio.com/bin/windows/base/R-3.2.0-win.exe then click “Save File” to save the file to your local disk.
(e) Mac users can go here http://cran.rstudio.com/bin/macosx/R-3.3.0.pkg then click “Save File” to save the file to your local disk.

(f) Once the installation file was saved execute it and select “Run”, then choose your language, say “English”, click “OK”, click “Next”, click “Next” again after reading the license info, choose an installation folder (default is just fine).

(g) You then need to choose the components you want to install. If you have enough space on your hard disk you can choose “Full installation”, which will require about 64.6MB.

(h) Click “Next”, click “Next” again to accept the default choice, choose the folder in which short-cuts should be created, click “Next”, **check all options**, click “Next”, and the installation will start. The whole process might run for anywhere from a few seconds to a few minutes depending on your computer. Click “Finish” after the setup finishes.

### 1.2 RStudio installation

RStudio is a wonderful complement of R. It eases many of the rather tedious tasks that R users have to face while looking at the white screen of R. RStudio is multi-platform and free.


2. Windows users can select http://download1.rstudio.org/RStudio-0.98.1103.exe

3. Mac users can select http://download1.rstudio.org/RStudio-0.98.1103.dmg

4. In either case the default options are fine.

5. If you installed R correctly you will be able to execute RStudio.

6. In RStudio select Tools, Pane Layout, and select Console and Source, then “Apply.”

### 1.3 notepad++

- Windows users may prefer notepad notepad++
- It also allows us to execute R code directly.
- It is very advanced in code highlighting and manipulation.
- If you decide to use this you should also download NppToR.exe which can be obtained from NppToR.exe
1.4 R & RStudio

- We will be using R through RStudio.
- RStudio will allow us to execute the R code to replicate the exercises seen throughout our course.
- Today we will familiarize ourselves with R and RStudio.
- There is no “unique right way to teach R,” but certainly there are wrong ways to do so.
- We will be looking at the simplest version of R today and then we will move to its fancy version which is dressed as the RStudio platform.

2 a < − “hello world”

(a) Launch R like any other program

(b) The working directory is the folder in which R reads data and saves output.

> #Assume we want to read data from our documents
> a<-read.csv("C:/Users/msgc/Documents/my.data.csv")

(c) You can see your working directory by typing

> getwd() #Stands for get working directory


(d) NOTE that R requires either a slash (/) or two backslash (\\) to recognize paths

(e) You can set the wd with the following command, which requires quotation marks (""")

> setwd("C:/Users/msgc/Documents/R")

(f) In R, no feedback is good, whenever you receive a feedback there is something wrong with the code or the data. For example try typing this in your R console

> setwd()

(g) R is an interactive environment with the command prompt, you typically just hit “Enter” to submit a command. In RStudio it will be “Ctrl+r” in windows and “command+r” in Mac.

(h) To exit type

> q()
(i) If you save your “workspace image” a file called “.RData” will be created under your working directory. If you double click this “.RData” you will have access to all your previous work. I do not recommend this procedure. It is better to save the commands used as we will do in this course.

2.1 Getting help

- If you know the name of a function, say t.test, and want to learn more details about it like syntax etc., type:

```r
> help(t.test)
> #or
> ?t.test
```

- If you don’t know the name of a command you can start a browser:

```r
> help.start()
```

2.2 Work-flow

- I would recommend editing all your commands for a project using your favorite text editor (I use Notepad++) which allows me to execute directly to the R console. Since there is no version of Notepad++ for Mac, we will use RStudio.

- Note that all R commands and objects are case-sensitive.

- Anything after # in a line is considered comment and is hence ignored by the system.

- This sequence <- assigns values to objects.

- Everything in R is an object.

- You could use the following command to execute all commands in a file, say “Function truncation.R”:

```r
> source("Function truncation.R")
```

- You could also let R to automatically save all output in a file, say, “homework1.txt”, using the following command before running any other command:

```r
> sink("homework1.txt")
```

- You could stop the automatic saving of output by the following command:

```r
> sink()
```
2.3 Number and Vectors

- Number variables:
  
  ```
  > a<-2
  > b<-3
  > a+2
  [1] 4
  > a+b
  [1] 5
  > a*3
  [1] 6
  > length(a)
  [1] 1
  ```

- A number is just a vector with length 1.

- Example of vectors and simple calculations:
  
  ```
  > a<-c(1, .5, 2.7, 9.8)
  > a
  [1] 1.0 0.5 2.7 9.8
  > length(a)
  [1] 4
  > a+3
  [1] 4.0 3.5 5.7 12.8
  > a*5
  [1] 5.0 2.5 13.5 49.0
  > 1/a
  [1] 1.0000000 2.0000000 0.3703704 0.1020408
  > sqrt(a)
  ```
1.0000000 0.7071068 1.6431677 3.1304952
> b<-exp(a)
> b
[1] 2.718282 1.648721 14.879732 18033.744928
> log(b)
[1] 1.0 0.5 2.7 9.8
> b<-10^a+5
> b
[1] 1.500000e+01 8.162278e+00 5.061872e+02 6.309573e+09
> log10(b-5)
[1] 1.0 0.5 2.7 9.8
> b<-c(1,2,3,4)
> a
[1] 1.0 0.5 2.7 9.8
> b
[1] 1 2 3 4
> a*b
[1] 1.0 1.0 8.1 39.2
> b<-c(0,1)
> a*b
[1] 0.0 0.5 0.0 9.8

- The command `ls()` allows us to see what we have currently available in our working directory

> ls()

[1] "a" "b" "c" "colors" "discrim"
2.4 Some commonly used functions

> length(a)
[1] 4

> sum(a)
[1] 14

> prod(a)
[1] 13.23

> mean(a)
[1] 3.5

> var(a)
[1] 18.52667

> #To calculate variance manually
> sum((a-mean(a))^2)/(length(a)-1)
[1] 18.52667

> max(a)
[1] 9.8

> min(a)
[1] 0.5

> range(a)
[1] 0.5 9.8

> summary(a)

  Min. 1st Qu.  Median    Mean  3rd Qu.     Max.
  0.500  0.875   1.850   3.500  4.475   9.800

> sort(a)
[1] 0.5 1.0 2.7 9.8

• Concatenating two vectors:
> a<-c(1,2,3)
> b<-c(6,7,8)
> c<-c(a,b)
> c

[1] 1 2 3 6 7 8

> #Order matters
> (c<-c(b,a))

[1] 6 7 8 1 2 3

• Generation of special sequences:

> a<-4:10
> a

[1] 4 5 6 7 8 9 10

> b<-10:4
> b

[1] 10 9 8 7 6 5 4

> a<-seq(5,10,by=0.5)
> a

[1] 5.0 5.5 6.0 6.5 7.0 7.5 8.0 8.5 9.0 9.5 10.0

> c<-seq(5,10,length=11)
> c

[1] 5.0 5.5 6.0 6.5 7.0 7.5 8.0 8.5 9.0 9.5 10.0

> rep(5,10)

[1] 5 5 5 5 5 5 5 5 5 5

> rep("blue",5)

[1] "blue" "blue" "blue" "blue" "blue"

> colors<-c(rep("blue",2),rep("red",2))
> colors

[1] "blue" "blue" "red" "red"

• Logical operators <, <=, >, >=, ==, !=
• Logical vectors can be generated by the logical operators

```r
> 1<5
[1] TRUE
> a<-1:10
> a<5
[1] TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE
> a<=5
[1] TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE
> a<=5 & a>2
[1] FALSE FALSE TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE
> a<2 | a>=8
[1] TRUE FALSE FALSE FALSE FALSE FALSE FALSE TRUE TRUE TRUE
```

```r
> b<- a<=5
> b
[1] TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE
> b+2.5
[1] 3.5 3.5 3.5 3.5 3.5 2.5 2.5 2.5 2.5 2.5
```

• Working with specific components of a vector:

```r
> a<-(-4):4
> a
[1] -4 -3 -2 -1 0 1 2 3 4
> a[2]
[1] -3
> a[2]<-0
> a
[1] -4 0 -2 -1 0 1 2 3 4
```
> a[a<0]<-0.5
> a
[1] 0.5 0.0 0.5 0.5 0.0 1.0 2.0 3.0 4.0
> a[4:8]
[1] 0.5 0.0 1.0 2.0 3.0
> b<-1:10
> b[-5]
[1]  1  2  3  4  6  7  8  9 10
> b[-(5:7)]
[1] 1 2 3 4 8 9 10

- Vectors with characters as components:

> a<-0:6
> a
[1] 0 1 2 3 4 5 6
> b<-as.character(a)
> b
[1] "0" "1" "2" "3" "4" "5" "6"
> b[b=="0"]<="zero"
> b
[1] "zero" "1" "2" "3" "4" "5" "6"
> a<-as.numeric(b)
> a
[1] NA  1  2  3  4  5  6
> #NA (not available) is used to designate missing data
> is.na(a)
[1]  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
> a+2
[1] NA  3  4  5  6  7  8
> a*2
[1] NA  2  4  6  8 10 12

> b<-a[!is.na(a)]
> b
[1] 1 2 3 4 5 6

• To remove all the objects in R without closing the session we can use

> rm(list=ls()) #removes everything
> rm(a,b) #removes specific objects, in this case a and b
> ls()

2.5 Matrices, Lists and Data Frames

• Generation and indexing of a matrix:

> a<-matrix(1:12,nrow=3,ncol=4, byrow=T)
> a

[1,] 1  2  3  4
[2,] 5  6  7  8
[3,] 9 10 11 12

> a<-matrix(1:12,nrow=4,ncol=3, byrow=F)
> a

[,1] [,2] [,3]
[1,] 1  5  9
[2,] 2  6 10
[3,] 3  7 11
[4,] 4  8 12

> a<-matrix(rbinom(16, 1, .5),nrow=4,ncol=4, byrow=F)
> a

[1,] 1  0  0  1
[2,] 0  0  1  0
[3,] 1  1  1  0
[4,] 1  0  0  0

> diag(a) #Extracting elements of a matrix
> diag(a)<-0 #no self-selection
> a

[1,] 0 0 0 1
[2,] 0 0 1 0
[3,] 1 1 0 0
[4,] 1 0 0 0

Note that R assigns the values to cells of a matrix in a “column-first” order as above

> a<-matrix(1:12,nrow=3,ncol=4, byrow=T)
> a

[1,] 1 2 3 4
[2,] 5 6 7 8
[3,] 9 10 11 12

> a[2,4] #should be 8
[1] 8

> a[2,c(2,4)] #returns the value of columns
[1] 6 8

> a[2,]
[1] 5 6 7 8

> a[,c(2,4)]

[,1] [,2]
[1,] 2 4
[2,] 6 8
[3,] 10 12

> a

[1,] 1 2 3 4
[2,] 5 6 7 8
[3,] 9 10 11 12

13
> a[1:2,1:2] #means: (1,1), (1,2), (2,1), (2,2)

        [,1] [,2]
[1,]  1   2
[2,]  5   6

> dim(a)
[1] 3 4

• Some simple calculations:

> a*2

[1,]   2   4   6   8
[2,]  10  12  14  16
[3,]  18  20  22  24

> b<-c(-1,0,1,2)
> a*b

[1,]  -1   4   3   0
[2,]   0  -6  14   8
[3,]   9   0 -11  24

> a %*% b # matrix multiplication

        [,1]
[1,]   10
[2,]   18
[3,]   26

> t(a) #transpose

       [,1] [,2] [,3]
[1,]   1   5   9
[2,]   2   6  10
[3,]   3   7  11
[4,]   4   8  12

• Building a matrix:

> a<-c(1,2)
> a
[1] 1 2
> b<-cbind(a,c(4,5))
> b

   a
[1,] 1 4
[2,] 2 5

> c<-c(3,6)
> c

[1] 3 6

> b<-rbind(b,c)
> b

   a       c
[1,] 1 4 3
[2,] 2 5 6

> b<-cbind(b,7:9)
> b

   a       c       d
[1,] 1 4 7 3
[2,] 2 5 8 6

• Eigenvalues and Eigenvectors:

> b<-eigen(b)
> b

$values
[1]  1.611684e+01 -1.116844e+00 -5.700691e-16

$vectors
     [,1]       [,2]       [,3]
[1,] -0.4645473 -0.8829060  0.4082483
[2,] -0.5707955 -0.2395204 -0.8164966
[3,] -0.6770438  0.4038651  0.4082483
The object b contains data of two different types. It is called a list. A list is an object consisting of an ordered collection of objects known as its components. The components don’t need to be of the same mode or type. Lists provide a convenient way to return the results of a mathematical or statistical computation, like in the eigenvalue-eigenvector calculation where the results consist of two components.

```r
> names(b)
[1] "values"  "vectors"

> length(b)
[1] 2

> class(b)
[1] "list"

> str(b)
List of 2
$ values : num [1:3] 1.61e+01 -1.12 -5.70e-16
$ vectors: num [1:3, 1:3] -0.465 -0.571 -0.677 -0.883 -0.24 ...

> b$values
[1] 1.611684e+01 -1.116844e+00 -5.700691e-16

> b$vectors
                    [,1]          [,2]          [,3]
[1,] -0.4645473 -0.8829060  0.4082483
[2,] -0.5707955 -0.2395204 -0.8164966
[3,] -0.6770438  0.4038651  0.4082483

> b[2]
$ vectors
                [,1]          [,2]          [,3]
[1,] -0.4645473 -0.8829060  0.4082483
[2,] -0.5707955 -0.2395204 -0.8164966
[3,] -0.6770438  0.4038651  0.4082483
```

SNA relies on lists and matrices to create graphs. Graph is the visual representation of a matrix or list.
A data frames is a matrix-like structure whose columns can be of different types. Many datasets for statistical analysis can be convenient described by data frames: each row represents an observational unit and each column designates a random variable. A data frame is also a list, with each column as a component. There are several ways to generate a data frame. The first is to specify each column/component separately:

```r
> a<-c(3,-1.5,-2.7,5,1.2)
> b<-list(data=a,n=length(a),mean=mean(a),std.dev=sqrt(var(a)))
> names(b)
[1] "data"  "n"       "mean"    "std.dev"
> b

$data
 [1,] 3.0 -1.5 -2.7 5.0 1.2
$n
 [1] 5
$mean
 [1] 1
$std.dev
 [1] 3.161487
```

A data frame is also a list, with each column as a component. There are several ways to generate a data frame. The first is to specify each column/component separately:

```r
> a<-data.frame(name=c("A","B","C"),age=c(25,68,49))
> a

  name age
1   A 25
2   B 68
3   C 49

> a<-data.frame(a,gender=c("Female","Male","Female"))
> a

  name age gender
1   A 25 Female
2   B 68   Male
3   C 49 Female

> a<-data.frame(a,PHW=c(27,42,25))
> a
```
name age gender PHW
1  A  25  Female  27
2  B  68  Male  42
3  C  49  Female  25

- We can also convert a matrix into a data frame.

```r
> b <- matrix(1:12, 3, 4)
> b

[1,]  1  4  7  10
[2,]  2  5  8  11
[3,]  3  6  9  12

> b <- data.frame(b)
> b

   X1 X2 X3 X4
1   1  4  7 10
2   2  5  8 11
3   3  6  9 12

> colnames(b) <- c("a1","b2","c3","d4")
> b

   a1 b2 c3 d4
1   1  4  7 10
2   2  5  8 11
3   3  6  9 12

> b$c3

[1] 7 8 9

> b[,3]

[1] 7 8 9

> dim(b)

[1] 3 4
2.6 Reading external files

- Instead of manually entering a data frame or matrix, we read them from a file.
- The example consists of observations on six variables for 52 tenure-track professors in a small college. The variables are:
  - sx = Sex, coded 1 for female and 0 for male
  - rk = Rank, coded 1 assistant, 2 associate professor, and 3 full professor
  - yr = Number of years in current rank
  - dg = Highest degree, coded 1 if doctorate, 0 if masters
  - yd = Number of years since highest degree was earned
  - sl = Academic year salary, in dollars.
- This file can be downloaded from the Internet or be stored in your computer

```r
> discrim <- read.table("http://data.princeton.edu/wws509/datasets/
+ salary.dat", header=T)
```
- R can import data from many different formats, for instance, let’s try reading the same data but from STATA.

```r
> library(foreign)
> discriStata <- read.dta("c:\Users\msgc\salary.dta")
```
- The most useful format is comma separated values (*.csv) to read it we use read.csv

```r
> discrimCSV <- read.csv("c:\Users\msgc\salary.csv")
```

2.7 Exploring data frames

- head and tail functions by default show the first 6 and last six rows of the dataset

```r
> head(discrim)

          sx rk yr      dg yd sl
1     male full 25 doctorate 35 36350
2     male full 13 doctorate 22 35350
3     male full 10 doctorate 23 28200
4 female full  7 doctorate 27 26775
5     male full 19 masters 30 33696
6     male full 16 doctorate 21 28516
```

```r
> tail(discrim)
```

2.8 Data Analysis

- Let’s plot the distributions of variables in the discrim dataset

  > hist(discrim$yr, col="firebrick1", main="")
  > hist(discrim$yd, col="darkgoldenrod2", main="")
  > hist(discrim$sl, col="cyan2", main="")

- For categorical variables we have

  > table(discrim$sx)
Figure 1: Distribution of continuous variables

- We can superimpose histograms as follows which renders the following figures

Figure 2: Male-Female salaries
> p1 <- hist(discrim[discrim$sx=="male",]$sl, main="")
> p2 <- hist(discrim[discrim$sx=="female",]$sl, main="")
> plot(p1, col=rgb(0,0,1,1/4), xlim=c(15000,40000), ylim=c(0,15),
+ xlab="Male and female salaries")  # first histogram
> plot(p2, col=rgb(1,0,0,1/4), xlim=c(15000,40000), ylim=c(0,15),
+ add=T)  # second

- The relationship between salary and gender and rank can be plotted with the boxplot command:

![Boxplot of Gender and Salary](image)

![Boxplot of Rank and Salary](image)

(a) Gender and Salary (b) Rank and Salary

Figure 3: Distribution of continuous variables per factors

- We can plot two numeric variables using

  > plot(discrim$yr, discrim$sl)  #output omitted

- For more about graphical procedures, please see chapter 12 of “An Introduction to R” by W. N. Venables, D. M. Smith, and the R Development Core Team by following this link.

2.9 Some basic inferential stats

- Now let's test if the salary measurements of males and females differ

  > t.test(discrim$sl[discrim$sx=="male"], discrim$sl[discrim$sx=="female"])

  Welch Two Sample t-test
data:  discrim$sl[discrim$sx == "male"] and discrim$sl[discrim$sx == "female"]
t = 1.7744, df = 21.591, p-value = 0.09009
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -567.8539 7247.1471
sample estimates:
 mean of x  mean of y
 24696.79  21357.14

• We could also use the formula interface to carry out the same test

> t.test(sl~sx,data=discrim)

Welch Two Sample t-test
data:  sl by sx
t = -1.7744, df = 21.591, p-value = 0.09009
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -7247.1471 567.8539
sample estimates:
 mean in group female  mean in group male
 21357.14  24696.79

• To fit a linear model with sl as response and both age and gender as predictors:

> lm(sl~sx+yr, data=discrim)

Call:
lm(formula = sl ~ sx + yr, data = discrim)

Coefficients:
 (Intercept) sxmale yr
 18266.9 -201.5 759.0

• To see more details of the linear regression fit we could use summary()

> lmfit<-lm(sl~sx+yr, data=discrim)
> summary(lmfit)

Call:
lm(formula = sl ~ sx + yr, data = discrim)

Residuals:
 Min 1Q Median 3Q Max
-11034.6 -3159.4 -651.8 3184.8 13706.0
Coefficients:

|                | Estimate | Std. Error | t value | Pr(>|t|)  |
|----------------|----------|------------|---------|----------|
| (Intercept)    | 18266.9  | 1247.7     | 14.640  | < 2e-16  ***|
| sxmale         | -201.5   | 1455.1     | -0.138  | 0.89     |
| yr             | 759.0    | 118.3      | 6.414   | 5.37e-08 ***|

---

Signif. codes: 0 *** 0.001 ** 0.01 * 0.05 . 0.1 1

Residual standard error: 4306 on 49 degrees of freedom
Multiple R-squared: 0.4911, Adjusted R-squared: 0.4704
F-statistic: 23.65 on 2 and 49 DF, p-value: 6.482e-08

To see the diagnostics of the model we can type

```r
> par(mfrow=c(2,2))
> plot(lmfit)
```

![Model diagnostics](image)

**Figure 4: Model diagnostics**
• The object “lmfit” is actually a list, so is the output of the summary() function

```R
> class(lmfit)
[1] "lm"
> names(lmfit)
[1] "coefficients" "residuals" "effects" "rank" "fitted.values"
[6] "assign" "qr" "df.residual" "contrasts" "xlevels"
[11] "call" "terms" "model"
> lmfit$coeff
(Intercept) sxmale yr
18266.8722 -201.4668 759.0138
> is.list(summary(lmfit))
[1] TRUE
> summary(lmfit)$coeff

                  Estimate Std. Error    t value  Pr(>|t|)  
(Intercept) 18266.8722 1247.6999 14.6404369 1.601221e-19 
sxmale     -201.4668 1455.1450 -0.1384514 8.904511e-01 
         759.0138 118.3363 6.4140410 5.366076e-08

> summary(lmfit)$coeff[,4]
(Intercept) sxmale yr
1.601221e-19 8.904511e-01 5.366076e-08
```

• The last command extracts the p-values out of the results as a vector.

2.10 Programming in R (Chapters 9 and 10)

• Commonly used control statements for programming:

```R
> if (expr_1) expr_2 else expr_3
> for (name in expr_1) expr_2
> while (condition) expr
```

• The following example truncates all numbers in the vector “a” at ±2:

```R
> a<-rnorm(100,0,1)
> summary(a)
```
> for (i in 1:100) {
+   if (a[i] > 2) a[i] <- 2 else if (a[i] < -2) a[i] <- -2
+ }
> summary(a)

> a <- rnorm(100, 0, 1)
> a[a > 2] <- 2
> a[a < -2] <- -2

• The codes above were intended to demonstrate the use of the conditional execution and the loop statements. A more efficient way to achieve the truncation is:

> a <- rnorm(100, 0, 1)
> a[a > 2] <- 2
> a[a < -2] <- -2

• We can write a function to do the truncation:

> tr <- function(x, limit)
+   { limit <- abs(limit)
+     x[x > limit] <- limit
+     x[x < -limit] <- -limit
+     x
+   }
> a <- tr(a, 2)
> a <- tr(a, 1)
> summary(a)

2.11 Packages (Chapter 13)

• All R functions and the built-in datasets are stored in packages. A package needs to be loaded first before its functions and datasets can be accessed. To save memory and time, R only loads a few commonly used packages when starting. To see all packages available on the local computer, use

> library()

• To load a package, say a package for SSNA, use

> library(statnet)
There are many user-contributed packages for R. Some (like rpart) are distributed with each binary distribution of R, but for most of them we need to download from CRAN http://CRAN.R-project.org/ and its mirrors, or other repositories such as Bioconductor http://www.bioconductor.org/. When connected to the Internet we can use

```
> install.packages()
> #and
> update.packages()
```

to install and update the packages.

Welcome to the R world!