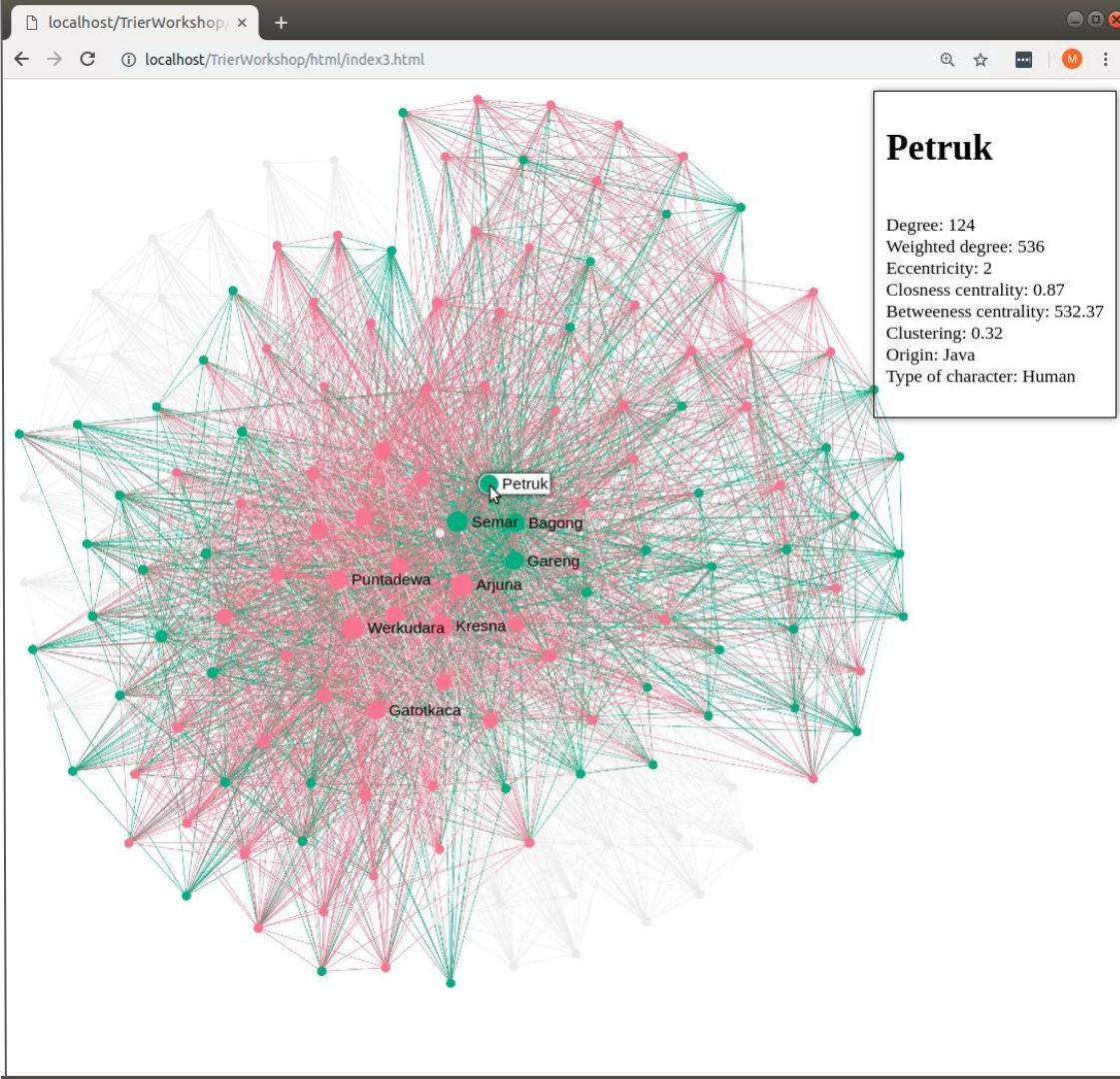


# Network analysis and DH

And an introduction to Gephi

Miguel Escobar Varela, PhD  
Assistant Professor, National University of Singapore  
[miguelescobar.com](http://miguelescobar.com)  
[@miguelJogja](https://twitter.com/miguelJogja)



# Objectives

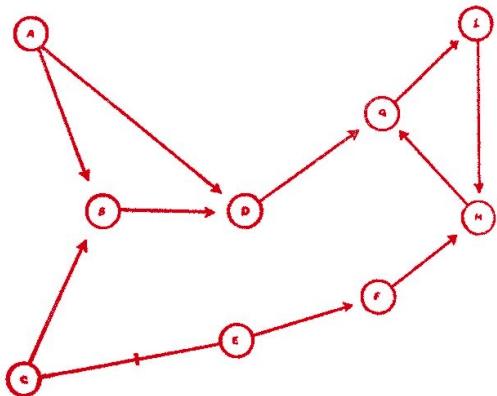
Describe key DH projects that use network analysis

Learn to use GEPHI to automatically calculate network-theoretical measurements and to create and export visualizations

# Part 1 Short overview of network analysis in DH

# Sociograms

- Jacob Levy Moreno



Moreno's Image of Who Recognized Whom Among a Collection of Babies (Moreno, 1934, p. 32).

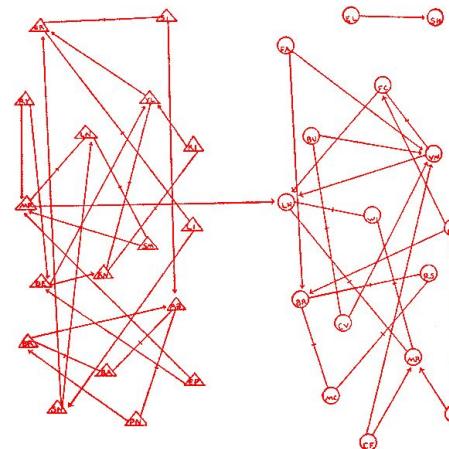
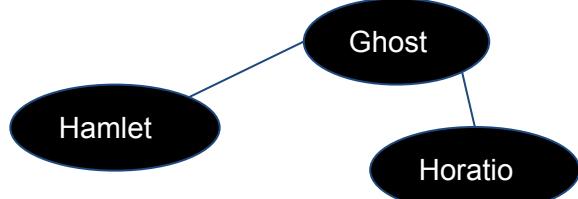


Figure 3. Friendship Choices Among Fourth Graders (from Moreno, 1934, p. 38).

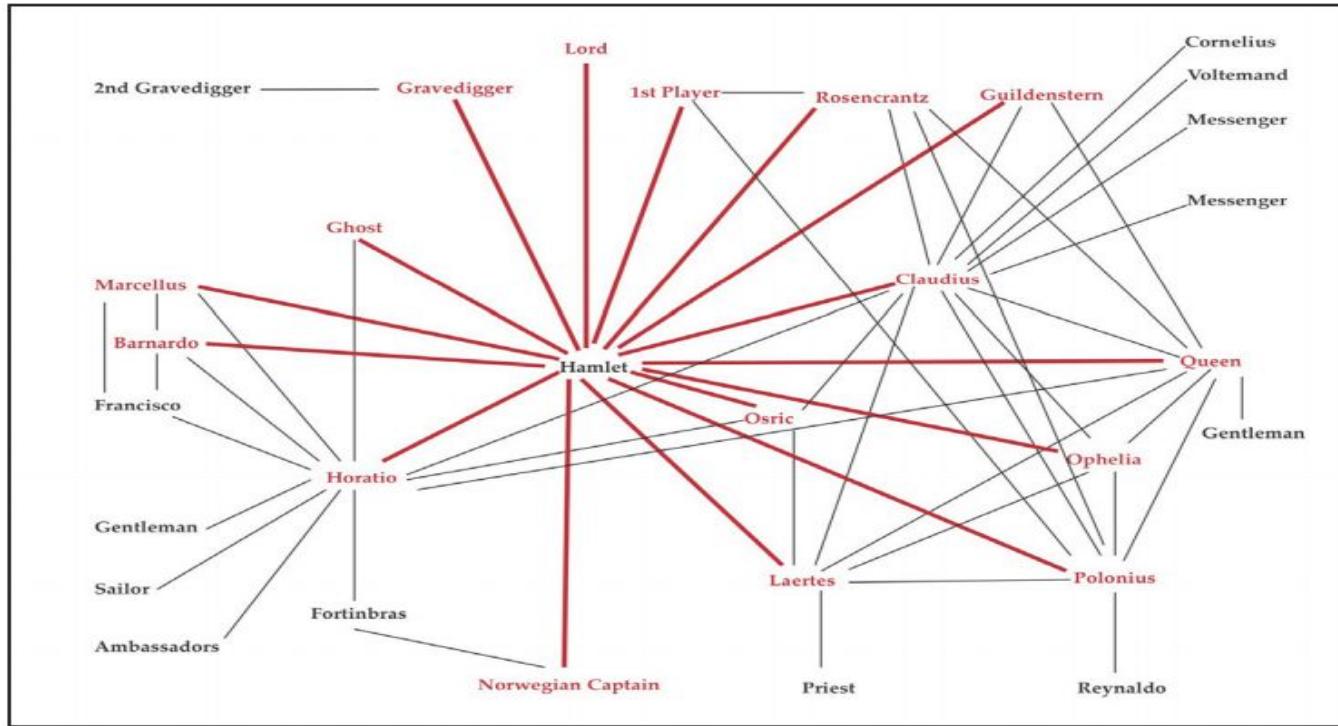
# Hamlet as a network

Characters as nodes, interactions as edges.



[This example shows only the edges for the Ghost, who only interacts with two other characters]

# Moretti on Hamlet



# Moretti on Hamlet

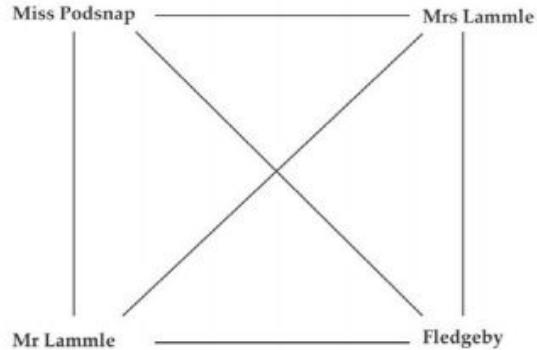
---

Macbeth shows a ‘region of death’ in the network. Except for Osric and Horatio, all characters linked to both Claudius and Hamlet are killed: “outside that region, no one dies in Hamlet [...] the tragedy is all there.” Moretti also identifies two separate components, sub-regions where each node is connected to every other node, which he interprets as corresponding the changing political landscape and the fight between the court and the state.

Moretti, *Network theory, plot analysis*, p272.

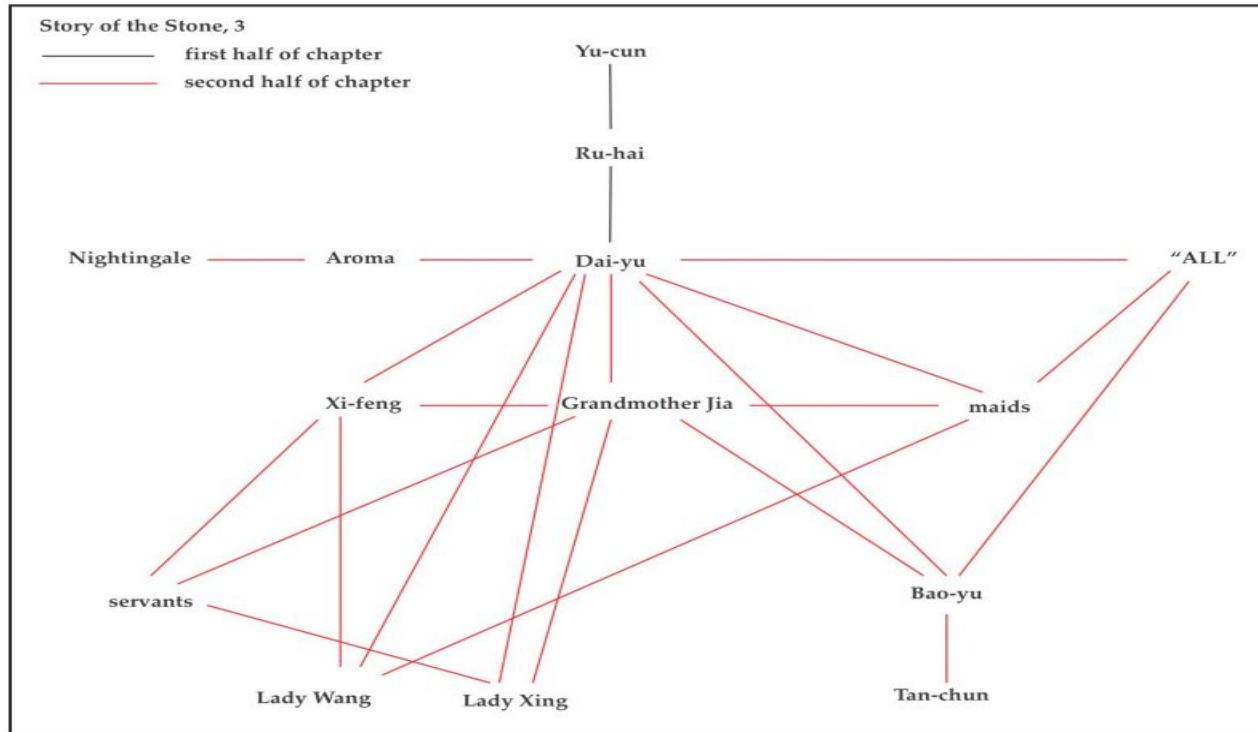
# Moretti on Comparisons

Our Mutual Friend II.4



Dickens' *Our Mutual Friend*

# Moretti on Comparisons



# Moretti on Comparisons

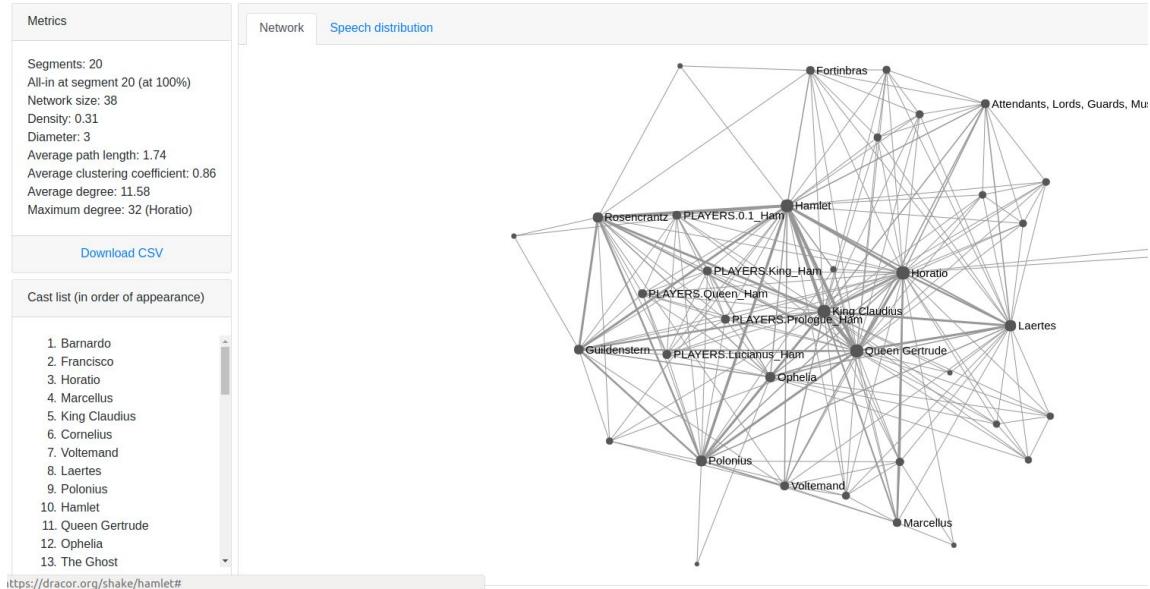
---

Dickens' *Our Mutual Friend* to the classical Chinese Cao Xueqin's *The Story of the Stone* (石头记), he suggest that the shape in the network of the Chinese text is determined by the concept of *guanxi* (关系), which can be understood as "connections" or a chain of "transactions that generate indebtedness." Ibid, 237.

# Analysis of Shakespeare networks

William Shakespeare

Hamlet



<https://dracor.org/shake>

# Network analysis of German drama

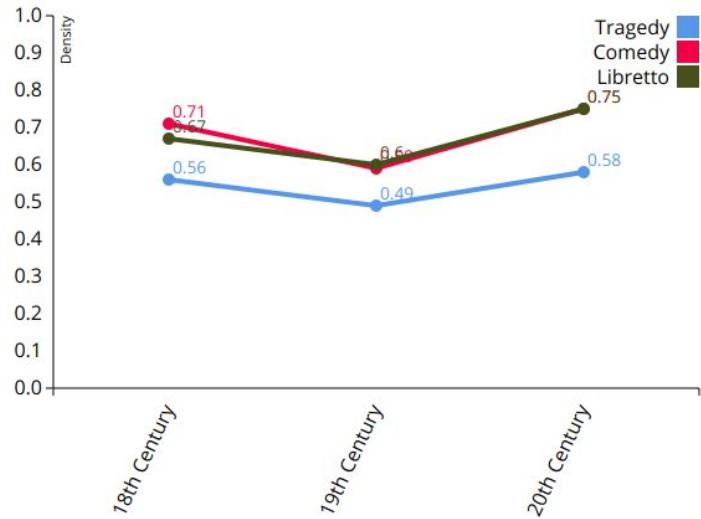
**Table 1: Network Measures, by Genre**

	N=	Number of Characters (Median)	Max Degree (Median)	Average Degree (Average)	Density (Average)	Average Path Length (Average)
Corpus	465	16	13	9,01	0,59	1,46
Tragedy	101	19	16	9,57	0,52	1,56
Comedy	92	14	11	8,61	0,67	1,36
Libretto	56	16	13,5	9,09	0,64	1,39
Other	216	17	14	8,88	0,59	1,48

<https://dlina.github.io/Network-Values-by-Genre/>

# Network analysis of German drama

**Fig. 5: Density (Mean), by Genre and Century**



<https://dlina.github.io/Network-Values-by-Genre/>

# Algee-Hewitt's work on dramatic networks

---

- Network properties of 3,568 English dramatic texts written 1550 and 1900 taken from ProQuest Literature Online drama corpus.
- Most probable recipient for each speech using a rule-based system (speech as directed edges) Eigenvector centrality (EC) and betweenness centrality (BC).
- BC can be used to estimate protagonism. Rather than merely identifying a protagonist, the distribution of BC across characters in a play shows how protagonism is shared among characters in a play, and how this changed over time, as plays are increasingly “less likely to feature a single central character (or cluster of characters) and more likely to depend on densely connected communities each featuring prominent characters in their own right: they are more likely to look like *A Midsummer Night’s Dream* than *Henry V*. ”
- BC shows what he calls ‘mediatedness’. Taking both measurements together, he identifies plays where the protagonist is the mediator and those where mediatedness and protagonism are shared by different characters.

# Algee-Hewitt's work on dramatic networks

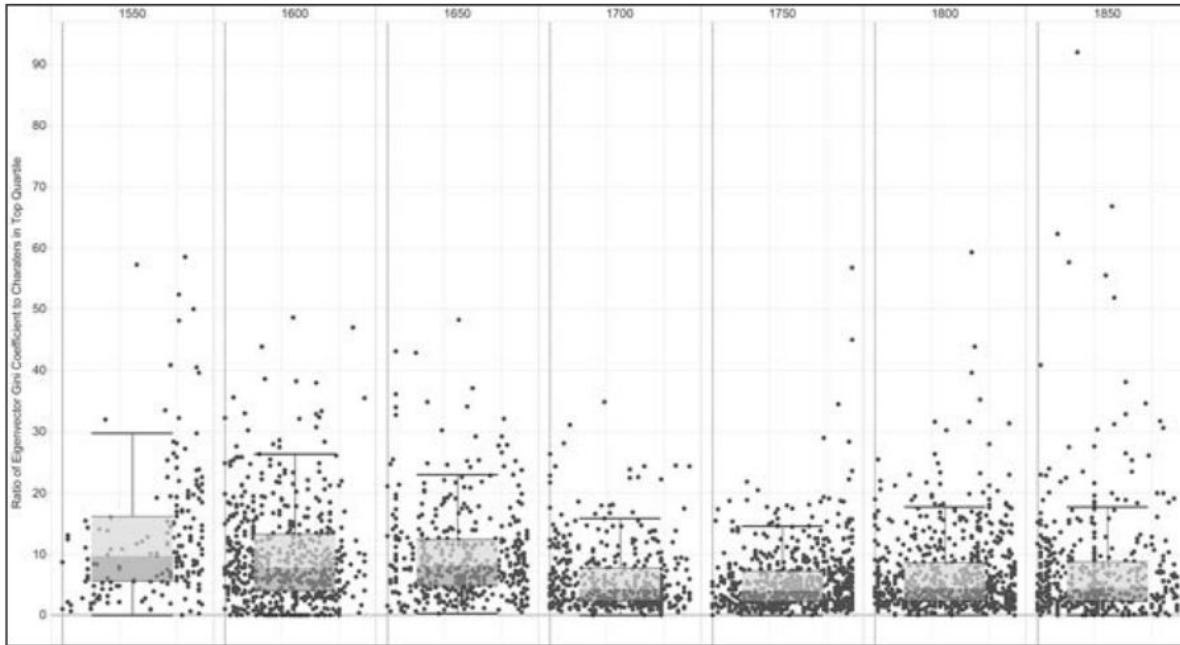


Fig. 4. Protagonism scores for plays in the Proquest corpus written between 1550 and 1899, divided into fifty-year increments.

Algee-Hewitt, Mark. 2017. "Distributed Character: Quantitative Models of the English Stage, 1550–1900." *New Literary History* 48 (4): 751–82.  
<https://doi.org/10.1353/nlh.2017.0038>.

# Algee-Hewitt's work on dramatic networks

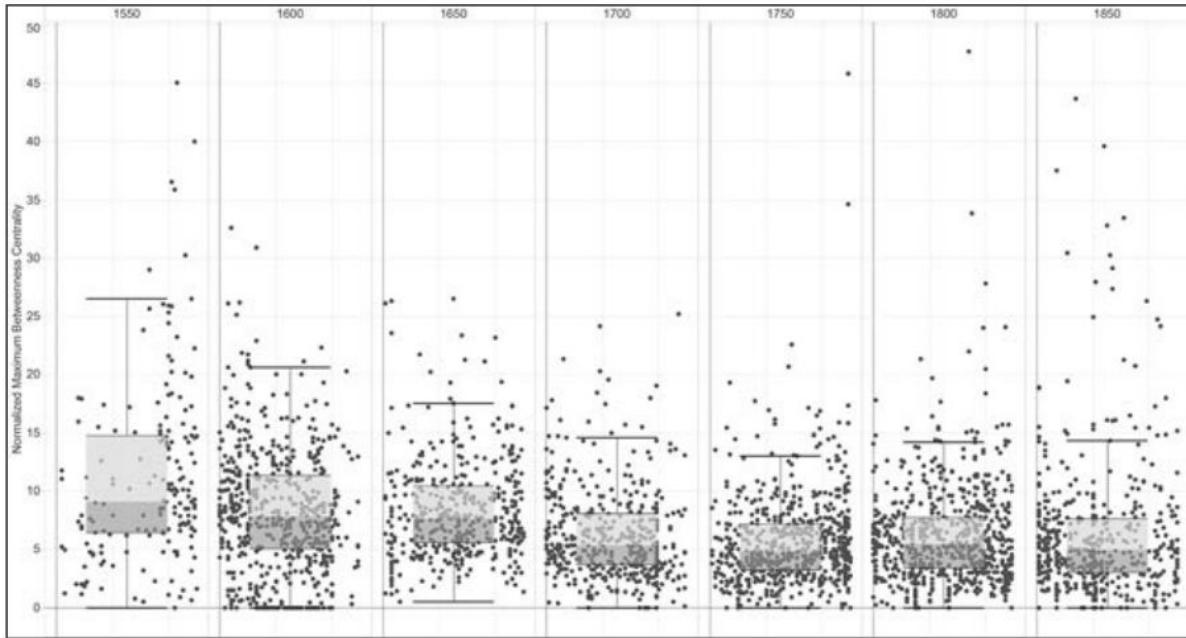


Fig. 7. Mediatedness scores for plays in the Proquest corpus written between 1550 and 1899, divided into fifty-year increments.

# Algee-Hewitt's work on dramatic networks

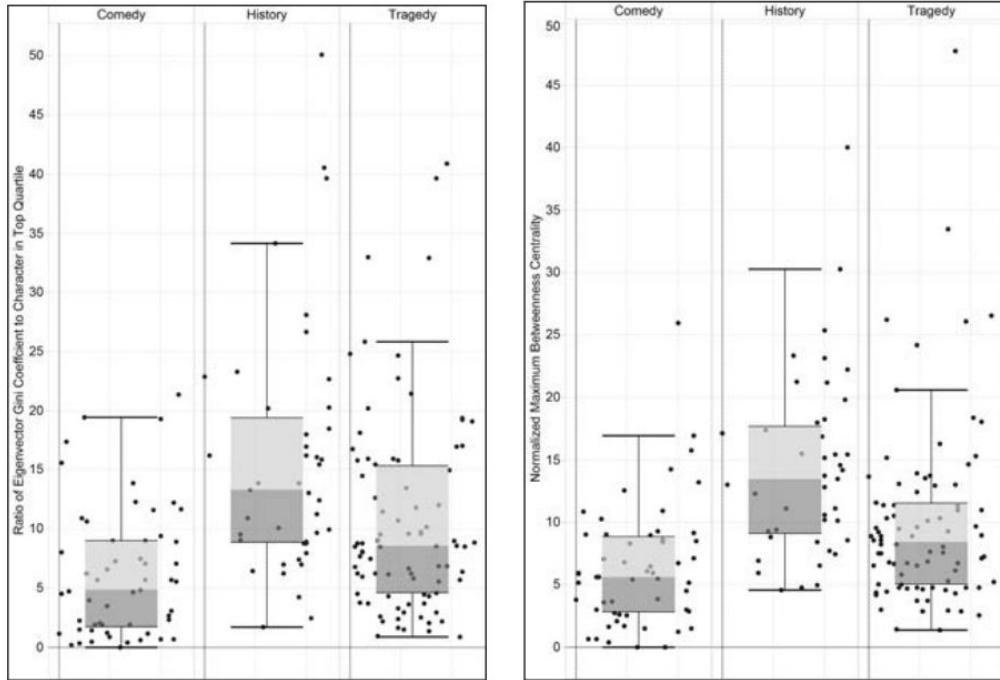


Fig. 8a and 8b. a) protagonism in a sample of comedies, histories, and tragedies; b) mediatedness in a sample of comedies, histories, and tragedies.

# Analysis of collaborations in theatre

The screenshot shows the homepage of the Australian Live Performance Database (AusStage). The top navigation bar includes the AusStage logo, a search bar, and links for Browse, Search, Learn, and Donate. A login link is also present. Below the navigation is a grid of colored cards representing different database categories:

Events	Contributors	Venues	Organisations	Genres	Functions	Exhibitions	
Map		Subjects		Works		Resources	International
<a href="#">How to contribute</a>							

Australian Live Performance Database,  
<https://www.ausstage.edu.au/pages/browse/>

# Three types of characters



Bima (Indian in origin)  
Indian characters 53%  
Avg. Wtd. Degree: 153.47

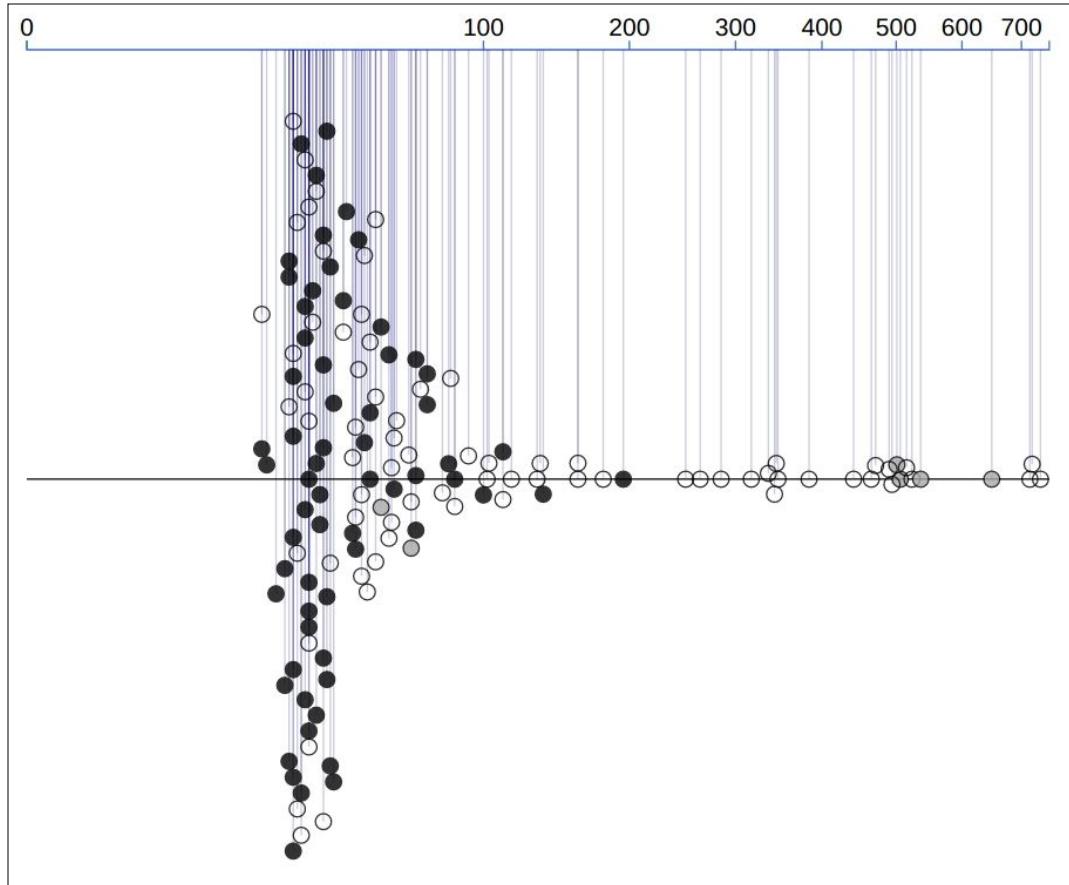


Dewa Ruci (Javanese in origin)  
Javanese Characters 47%  
Avg Wtd Degree 73.36



Gareng (Javanese  
clown-servant)

Although the percentage of Javanese and Indian characters is comparable, Javanese characters have disproportionately lower degrees.



The different characters according to their weighted degree. The x axis is in an exponential scale (factor = 0.3). Black circles are Javanese, white are Indian and gray are Javanese Punokawan.

Puppet representing  
Abimanyu.

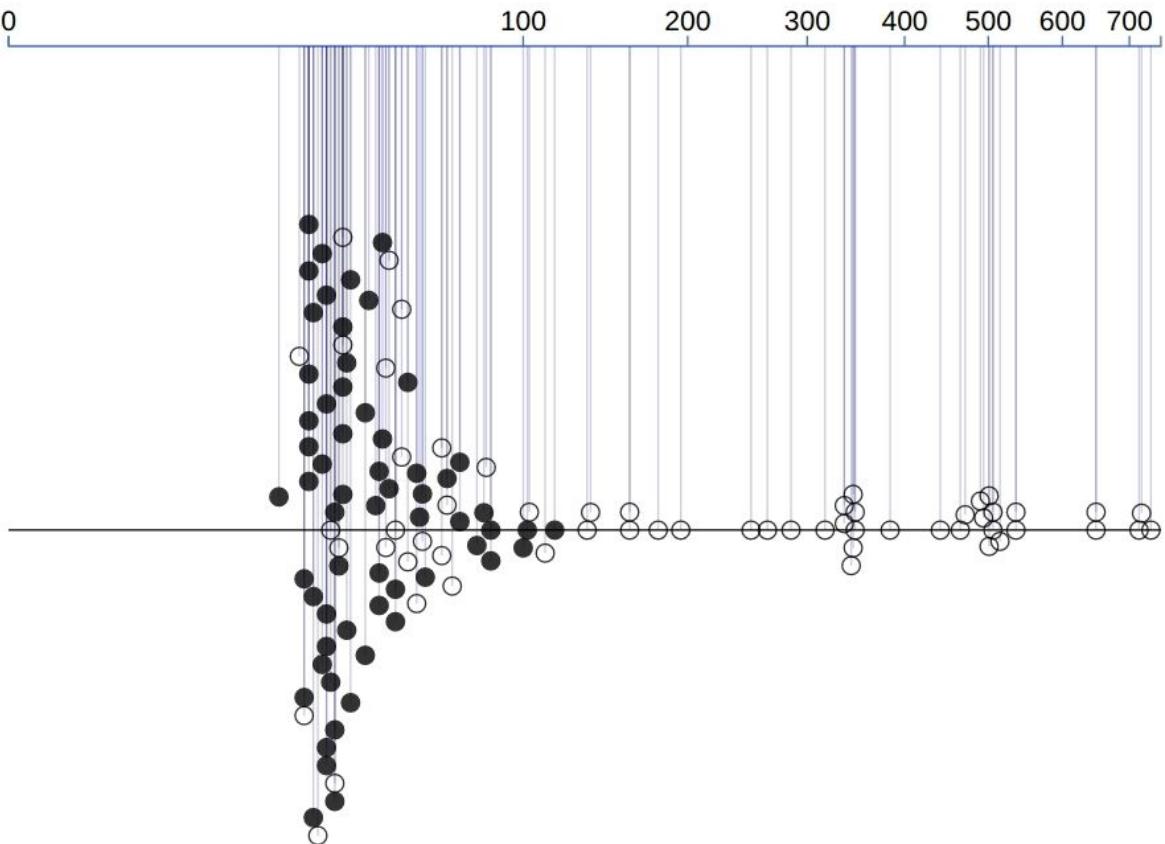


## What to do if you don't have enough puppets?

Abimanyu must always be represented by this puppet.

But other characters, such as Angganjani could also be represented by this puppet

Characters with lower degrees are interchangeable in performance, high degree characters are not.



The different characters according to their weighted degree. The x axis is in an exponential scale (factor = 0.3). Black circles are those where puppet interchange is permissible and white circles are those where it is not.

# Part 2 Working with Gephi

# Using Gephi

Which data to use?

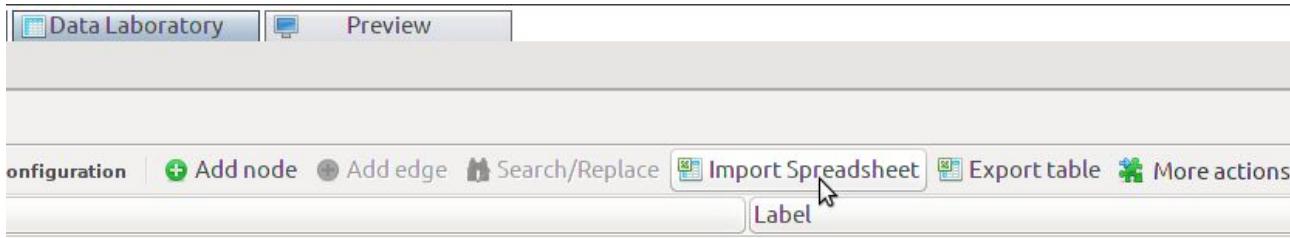
Loading data [my wayang data]. Loading edgelists and attributes.

Visualizations. Calculate basic features.

Arrange the project visually.

Export as GEXF

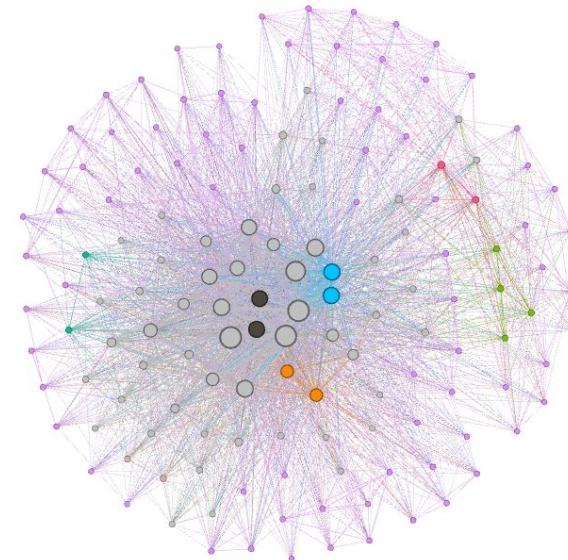
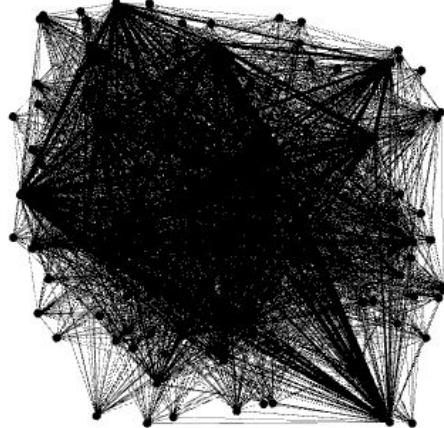
# Importing the data



Import gephi/data/wayang\_edgeList.csv

Make sure you include 'weight' and 'type' data

# Visualizing the network



We can apply some rules for color, spatial distribution and size to make the network diagram easier to understand.

# Let's run some statistics

## Use of these measurements

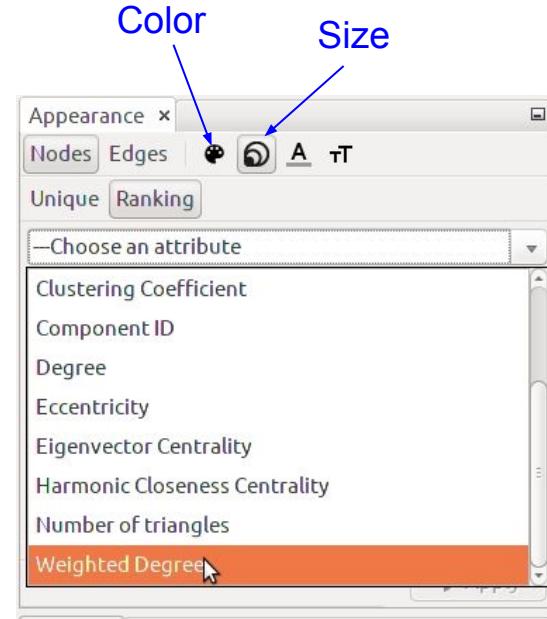
- They will help us better understand the network.
- Useful for styling (visual rules based on measurements).
- Will be eventually exported into our web project.

The screenshot shows a software window titled "Statistics" with a "Filters" tab selected. Under the "Settings" section, there is a checked checkbox labeled "Network Overview". Below this, a list of network metrics is displayed, each with a "Run" button and a circular progress indicator:

Metric	Run	Status
Average Degree	Run	Idle
Avg. Weighted Degree	Run	Idle
Network Diameter	Run	Idle
Graph Density	Run	Idle
HITS	Run	Idle
Modularity	Run	Idle
PageRank	Run	Idle
Connected Components	Run	Idle

# Styling the diagram

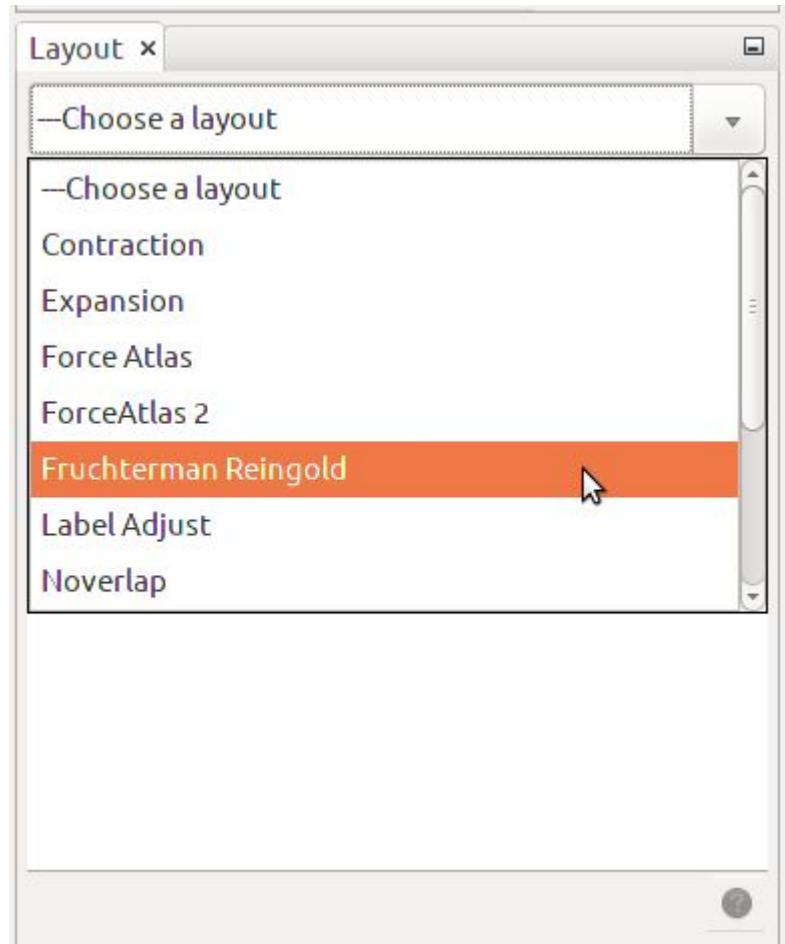
We can choose color and size parameters based on the measurements we calculated



# Automating the layout

Different algorithms can help with the spatial arrangement of the nodes

\* This is purely for visual convenience and the arrangement has no network-theoretical meaning



# Adding text labels

- 1) In the **nodes** table of the data laboratory tab, copy the ID to the Label column
- 2) Back in overview tab, select “Show node labels”

1

0.578767	0.0
0.657534	37.469238
0.811111	01.644649
0.578767	0
0.705556	5.959304
0.578767	0
0.578767	0
0.578767	0
0.578767	0
0.865556	54.586041
0.582222	0
0.678889	0.83832
0.924658	532.372892
0.613014	3.174418

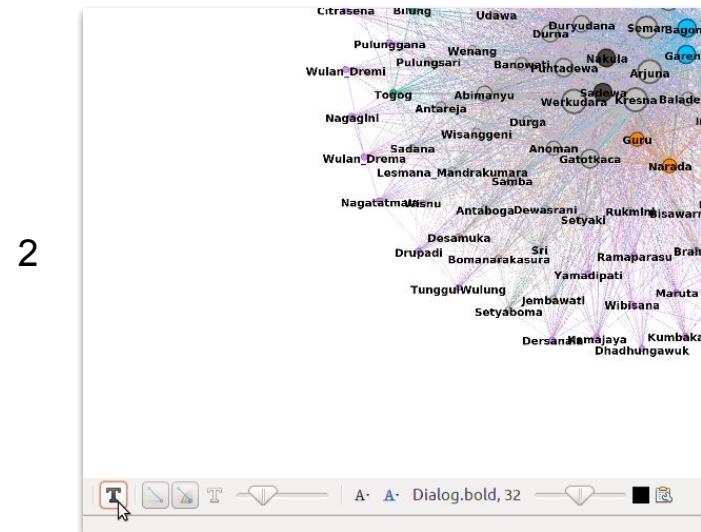
Copy data to other colu... X

Copy data from 'Id'

Copy to:

Label

Ok Cancel

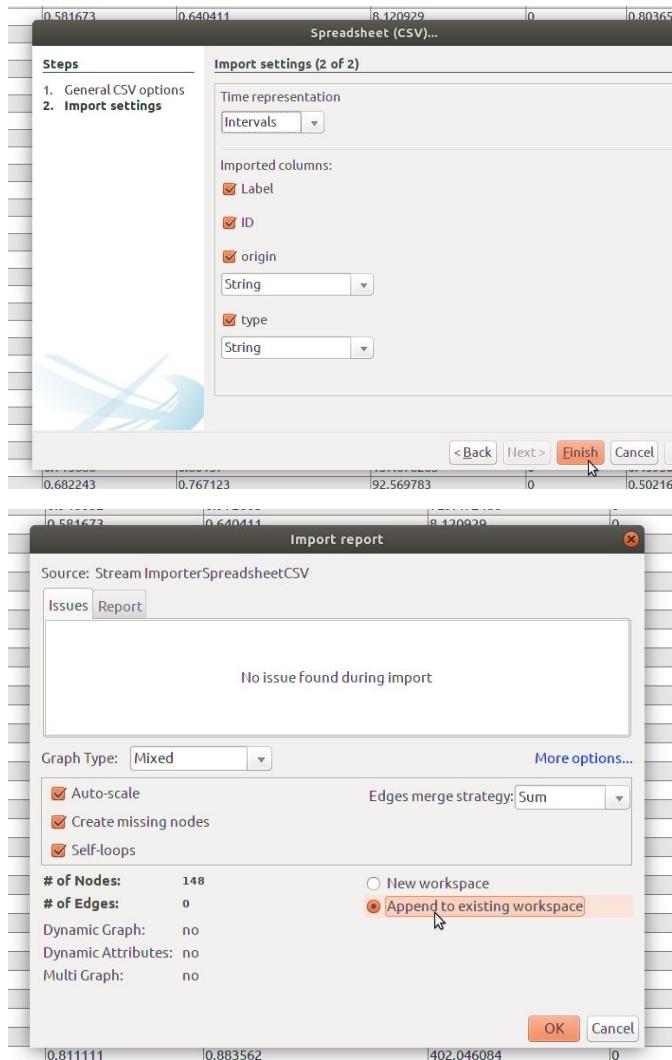


# Importing additional metadata

In the Data Laboratory tab, Nodes table, click “Import Spreadsheet” and select  
gephi/data/wayang\_nodeInfo.csv

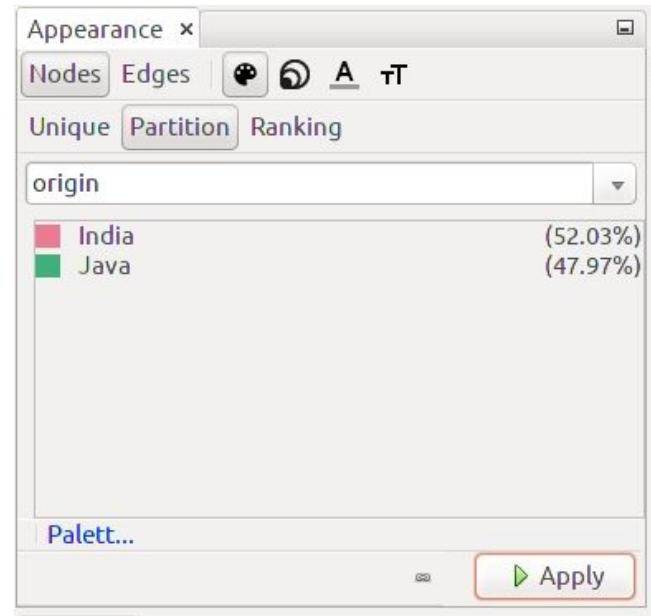
Select all the available columns.

Make sure you select “append to existing workspace”.



# Styling with additional metadata

We can determine the node color based on the origin of the wayang characters (India or Java)

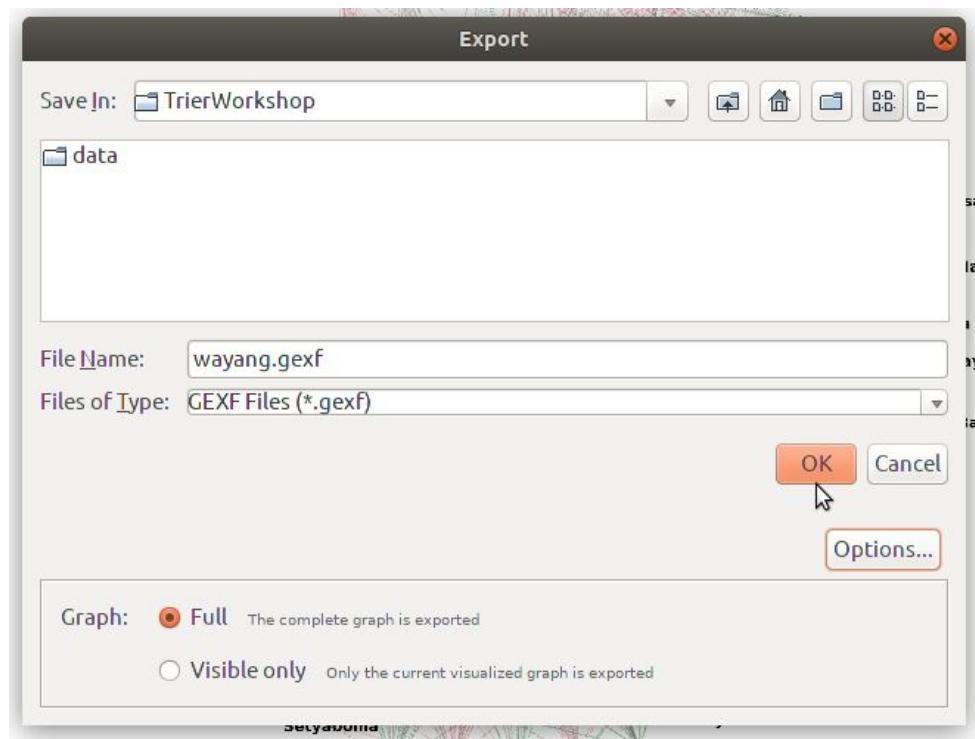


# Exporting the data

File -> Export -> Graph file...

Make sure you choose the GEXF format.

Save it inside html/data/ (different from gephi/data/)



# Part 3 Creating an interactive web visualization

# Sigma.js

<http://sigmajs.org/>



Sigma is a JavaScript library **dedicated to graph drawing**. It makes easy to publish networks on Web pages, and allows developers to integrate network exploration in rich Web applications.

## Get started with sigma

The following example shows how simple it is to use sigma to display a JSON encoded graph file.

DATA	HTML	RESULT
<pre>{   "nodes": [     {       "id": "n0",       "label": "A node",       "x": 0,       "y": 0,       "size": 3     },     {       "id": "n1",       "label": "Another node",       "x": 3,       "y": 0     }   ],   "edges": [     {       "source": "n0",       "target": "n1"     }   ] }</pre>	<pre>&lt;html&gt; &lt;head&gt; &lt;style type="text/css"&gt;   #container {     max-width: 400px;     height: 400px;     margin: auto;   } &lt;/style&gt; &lt;/head&gt; &lt;body&gt; &lt;div id="container"&gt;&lt;/div&gt; &lt;/body&gt; &lt;/html&gt;</pre>	A small network graph visualization showing two nodes, labeled 'A node' and 'Another node', connected by a single edge. The nodes are red circles, and the edge is a red line.

# Structure of the gexf parser

The gexf method takes three arguments:

- 1) the URL of the gexf file
- 2) An object with:
  - a) The ID of the HTML container for the graph
  - b) An object with additional settings
- 3) Additional functions

```
sigma.parsers.gexf(  
  'yourFile.gexf',  
  {  
    container: 'ID of html container',  
    settings: {/*additional settings*/}  
  },  
  function(s) {/*additional functions*/}  
);
```

# Loading local files

**Option 1:** Use your localhost, if you have configured this

**Option 2:** Upload to a remote server (i.e., GitHub pages)

**Option 3:** Run your browser with flags:

Windows:

Run -> `chrome.exe --allow-file-access-from-files`

MAC:

Open terminal -> `open /Applications/Google\ Chrome.app --args --allow-file-access-from-files`

\*Make sure you close all instances of Chrome before doing this

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <style>
5 </style>
6 </head>
7
8 <body>
9
10 <div id="sigma-container"></div>
11 <div id="infoWindow"></div>
12
13 <script src="js/jquery.js"></script>
14 <script src="js/sigma.js"></script>
15 <script src="js/sigma.parsers.gexf.min.js"></script>
16
17 <script>
18
19 //Additional method to determine a node's neighbors
20 sigma.classes.graph.addMethod('neighbors', function(nodeId) {
21   var k,
22     neighbors = {},
23     index = this.allNeighborsIndex[nodeId] || {};
24
25   for (k in index)
26     neighbors[k] = this.nodesIndex[k];
27   return neighbors;
28 });
29
30 //This is where we load the Gephi file
31 sigma.parsers.gexf(
32   , _____
33   { // Here is the ID of the DOM element that will contain the graph:
34     container: '_____',
35     settings: {
36       minNodeSize: 2,
37       maxNodeSize: 10,
38     }
39   },
40   function() {
41 }
```

# Sample html file

Open html/index.html in your chosen editor

Point to the correct data file and ID of the HTML element

Your data file: data/wayang.gexf

The container id: sigma-container

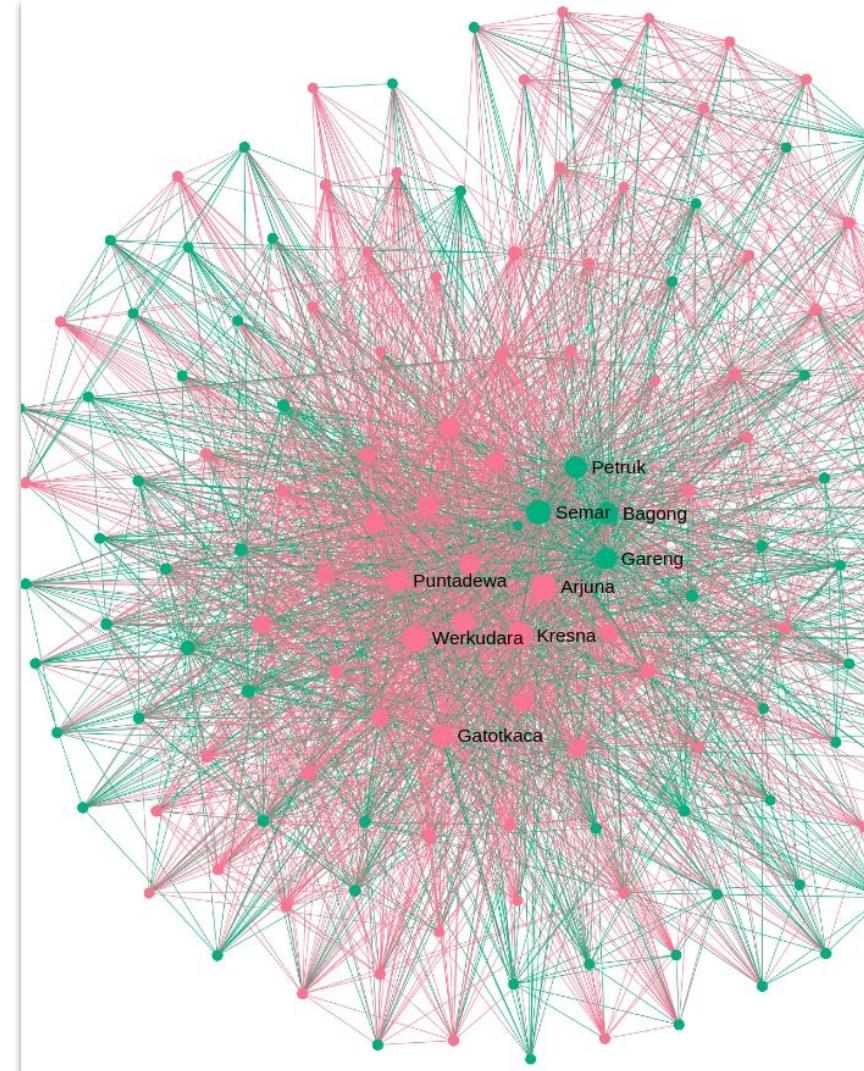
# The diagram isn't visible!

Make sure you give it dimensions via CSS

For example:

```
#sigma-container{  
    position: relative;  
    width: 800px;  
    height: 800px;  
}
```

See working file at [html/index2.html](#)



# Additional settings

<https://github.com/jacomyal/sigma.js/wiki/Settings>

<code>defaultNodeColor</code>	String	#000	The default color of nodes
<code>defaultLabelSize</code>	String	14	The default size of text used to draw node labels
<code>edgeColor</code>	String	"source"	When no <code>color</code> property is defined on the edge, determines how to determine the color. " <code>source</code> " and " <code>target</code> " use that node's color, respectively, while " <code>default</code> " just uses the <code>defaultEdgeColor</code> setting.
<code>minArrowSize</code>	Number	0	Defines the minimal edge's arrow display size.
<code>font</code>	String	"arial"	Defines the default font
<code>fontStyle</code>	String	""	A CSS descriptor for drawing the chosen font
<code>labelColor</code>	String	"default"	If set to " <code>node</code> ", the label will be drawn as the same computed color for the node. Otherwise, the default label color is used.
<code>labelSize</code>	String	"fixed"	Indicates how to choose the label's size. Available values: " <code>fixed</code> ", " <code>proportional</code> ".
<code>labelSizeRatio</code>	Number	1	The ratio between the font size of the label and the node size.
<code>labelThreshold</code>	Number	8	The minimum size a node must have on screen to see its label displayed. This does not affect hovering behavior.
<code>webglOversamplingRatio</code>	Number	2	The oversampling factor used in the WebGL renderer.

# Adding interactivity

Add the code just before s.refresh();

```
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
    //colors for the edges included in the graph
    s.graph.edges().forEach(function(e) {
      if (toKeep[e.source] && toKeep[e.target]) {
        e.color = e.originalColor;
      } else{
        e.color = '#eee';
      }
    });
    //display additional information
    explanation = "<p><h1>" + e.data.node_name + "</h1></p>" +
    explanation += "<p><br/>Degree: " + e.degree + "</p>" +
    explanation += "<br/>Weighted degree: " + e.weighted_degree +
    explanation += "<br/>Eccentricity: " + e.eccentricity +
    explanation += "<br/>Closeness centrality: " + e.closeness_centrality +
    explanation += "<br/>Betweenness centrality: " + e.betweenness_centrality +
    explanation += "<br/>Clustering: " + e.clustering +
    explanation += "<br/>Origin: " + e.data.origin +
    explanation += "<br/>Type of character: " + e.data.character +
    $("#infoWindow").html(explanation);
    // Since the data has been modified,
    // call the refresh method to make the changes effective.
    s.refresh();
  });
});
```

# Adding interactivity

- To navigate the XML file:  
`e.data.node.attributes.YourAttribute` or `e.data.node.attributes["Your Attribute"]`
- You can reference measurements calculated by Gephi as well as metadata that you imported
- Use the JavaScript function `.toFixed(n)` to limit the numbers to n decimal places
- Use the jQuery function `$("#yourDivID").html(yourVariable)` to display this additional information

```
//display additional information
explanation = "<p><h1>" + e.data.node.label +"</h1>";
explanation += "<p><br/>Degree: " + e.data.node.attributes.degree;
explanation += "<br/>Weighted degree: " + e.data.node.attributes["weighted degree"];
explanation += "<br/>Eccentricity: " + e.data.node.attributes.eccentricity;
explanation += "<br/>Closeness centrality: " + parseFloat(e.data.node.attributes.closenesscentrality).toFixed(2);
explanation += "<br/>Betweenness centrality: " + e.data.node.attributes.betweenesscentrality.toFixed(2);
explanation += "<br/>Clustering: " + e.data.node.attributes.clustering.toFixed(2);
explanation += "<br/>Origin: " + e.data.node.attributes.origin;
explanation += "<br/>Type of character: " + e.data.node.attributes.type;
$("#infoWindow").html(explanation);
```

# Visualizing the data as tables

<https://datatables.net/>

The screenshot shows the official website for DataTables. At the top, there are navigation links for 'DataTables' (with a circular icon), 'Editor' (with a circular icon), 'Manual', 'Download', 'Login / Register', and a search bar. The main heading is 'Add advanced interaction controls to your HTML tables *the free & easy way*'. Below this, three steps are outlined:

- 1 - Include these two files ↴  
CSS //cdn.datatables.net/1.10.19/css/jquery.dataTables.css  
JS //cdn.datatables.net/1.10.19/js/jquery.dataTables.js
- 2 - Call this single function ↴  
1 \$(document).ready( function () {  
2 \$('#myTable').DataTable();  
3 } );
- 3 - You get a fully interactive table →

A large teal button at the bottom says 'Full Getting Started Guide'.

To the right, a preview of the DataTables library is shown. It features a table with 10 entries per page, a search bar, and various sorting and filtering options. The table data is as follows:

Name	Position	Office	Age	Start date
Airi Satou	Accountant	Tokyo	33	2008/11/28
Angelica Ramos	Chief Executive Officer (CEO)	London	47	2009/10/09
Ashton Cox	Junior Technical Author	San Francisco	66	2009/01/12
Bradley Greer	Software Engineer	London	41	2012/10/13
Brenden Wagner	Software Engineer	San Francisco	28	2011/06/07
Brielle Williamson	Integration Specialist	New York	61	2012/12/02
Bruno Nash	Software Engineer	London	38	2011/05/03
Caesar Vance	Pre-Sales Support	New York	21	2011/12/12
Cara Stevens	Sales Assistant	New York	46	2011/12/06
Cedric Kelly	Senior Javascript Developer	Edinburgh	22	2012/03/29

Below the preview, it says 'Showing 1 to 10 of 57 entries' and includes a navigation bar with buttons for 'Previous', page numbers 1 through 6, and 'Next'.

# Adding interactive tables

Export table from Gephi as .csv file

Modify the code in table.html to load your CSV file as a JSON object

```
7 <script src="js/jquery.js"></script>
8 <script src="js/dataTables.min.js"></script>
9
10 <table id="dataTable" class="display" cellspacing="0" width="100%">
11   <thead>
12     <tr>
13     </tr>
14   </thead>
15 </table>
16
17 <script>
18 //container JSON object
19 var json = {"data":{}};
20
21 $.ajax({
22   url: "data/example.csv", //the name of the csv file
23   async: false,
24   dataType: "text",
25   success: function (csv) {
26     toJSON(csv);
27   }
28 });
29
30 function toJSON (data){
31   lines = data.match(/[\^r\n]+/g);
32   //populate the array
33   dataArray = []
34   var i;
35   for (i= 1; i<lines.length; i++){
36     dataArray.push(lines[i].split(","));
37   }
38   json["data"] = dataArray;
39   //make the table
40   headers = lines[0].split(",");
41   for (j in headers){
42     $('<th></th>').text(headers[j]).appendTo($("#dataTable>thead>tr"));
43   }
44   table = $("#dataTable").DataTable(json);
45 }
46
47 </script>
48 </body>
49 </html>
```

# Additional exercise

Do you have any network data to visualize interactively?

You can download data from this source: <https://dracor.org/>

# Troubleshooting

On Ubuntu 18.04 default Java is installed via packages: default-jre, default-jre-headless

These depend on: openjdk-11-jre, openjdk-11-jre-headless, which are Java 10.

With such packages installed, no charts are plotted, and the Options > Visualizations > OpenGL page shows no detail of graphic card and drivers.

Reverting back to Java 8 restores Gephi functionality:

```
sudo apt remove openjdk-11-jre openjdk-11-jre-headless
```

```
sudo apt install openjdk-8-jre
```

# References

- Algee-Hewitt, Mark. 2017. "Distributed Character: Quantitative Models of the English Stage, 1550–1900." *New Literary History* 48 (4): 751–82. <https://doi.org/10.1353/nlh.2017.0038>.
- Moretti, Franco. 2011. "Network Theory, Plot Analysis." *New Left Review*, no. 68 (March): 80.
- Fischer, Frank, Matthias Göbel, Dario Kampkaspar, Christopher Kittel, and Peer Trilcke. 2017. "Network Dynamics, Plot Analysis: Approaching the Progressive Structuration of Literary Texts." In *Digital Humanities 2017 Conference Abstracts*. Montreal, Canada.
- Fischer, Frank, Peer Trilcke, Carsten Milling, and Daniil Skorinkin. 2018. "To Catch A Protagonist: Quantitative Dominance Relations In German-Language Drama (1730–1930)." In *Digital Humanities 2018 Conference Abstracts*. Mexico City, Mexico.
- Trilcke, Peer, and Frank Fischer. n.d. "Dracor.Org." Accessed September 5, 2018. <https://dracor.org/>.
- Trilcke, Peer, Frank Fischer, Mathias Göbel, and Dario Kampkaspar. 2016. "Theatre Plays as 'Small Worlds'? Network Data on the History and Typology of German Drama, 1730–1930." In *Digital Humanities 2016: Conference Abstracts*, 385–87. Krakow, Poland: agiellonian University & Pedagogical University. <https://dlina.github.io/Our-Talk-at-DH2015/>.
- . n.d. "Comedy vs. Tragedy: Network Values by Genre." *DLINA: Digital Lintrary Analysis* (blog). Accessed August 24, 2018. <https://dlina.github.io/Network-Values-by-Genre/>.
- Trilcke, Peer, Frank Fischer, and Dario Kampkaspar. 2015. "Digital Network Analysis of Dramatic Texts." In *Digital Humanities 2015 Conference Abstracts*. Sydney, Australia.