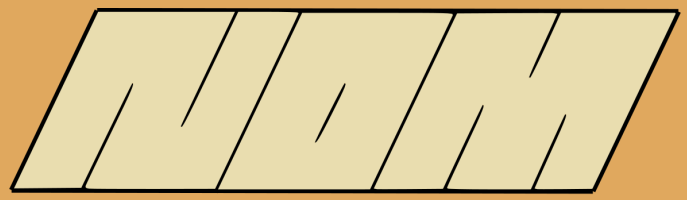
This project is funded and supported by:

THIS.Institute NIHR | Applied Research Collaboration South West Peninsula UNIVERSITY OF EXETER



Network-based Operational Modelling

Advanced network analysis session 2
Graph Visualisation

Dr Sean Manzi - University of Exeter, UK

Health Service Modelling Associates Programme 2020

Session structure

- 0930 – 1000 How graph visualisations work
- 1000 – 1030 Standardised algorithms and predefined coordinate systems
- 1030 – 1100 Visualising graphs with NetworkX and Holoviews
- 1100 – 1115 Comfort break
- 1115 – 1130 Visualising graphs with Gephi
- 1130 – 1215 Task time
- 1215 – 1230 Round up

Learning objectives

- Understand how graph algorithms seek to represent data
- Know how to visualise graphs using a variety of open source approaches
- Be able to customise a graph visualisation to improve its usefulness

How graph visualisation works

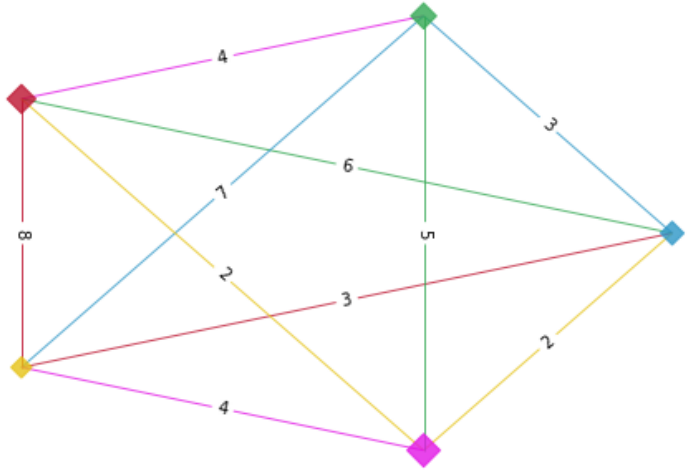
Network graphs provide a flexible way to visualise multiple attributes of the data. The three main components of the graph that can be customised to represent different aspects of the data are:

- Nodes
- Edges
- Layout

How graph visualisation works

Node customisation

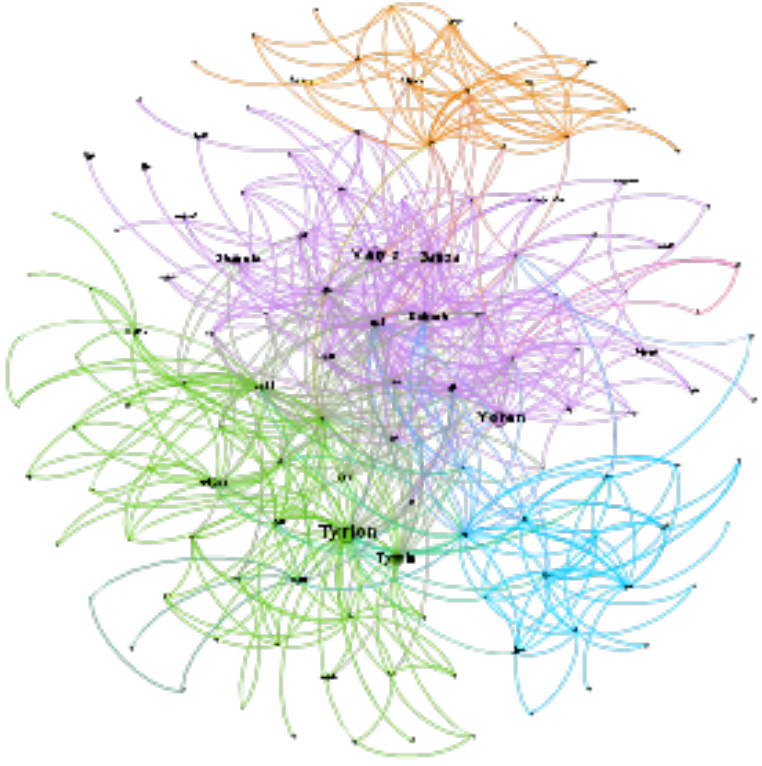
- Size - for continuous attributes
- Colour - for discrete attributes
- Shape - for discrete attributes
- Opacity - for continuous attributes



How graph visualisation works

Edge customisation

- Thickness - for continuous attributes
- Colour - for discrete attributes

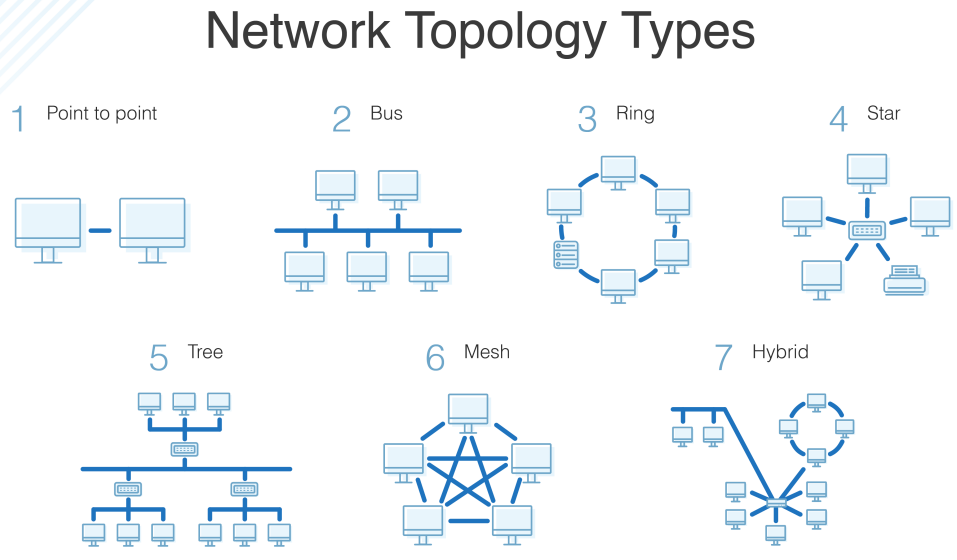


How graph visualisation works

Layout customisation

The basic layout of the graph is determined by the structure of the network, also called the network topology. Some common network structures are:

- Heirarchical
- Tree/Forest
- Ring
- Mesh
- Star
- Bus



How graph visualisation works

Simple and highly structured systems might conform to a simple topology. Complex systems are often a hybrid of two or more topologies making them more difficult to visualise. There are algorithms that help to visualise complex networks with non-standard topologies.

How graph visualisation works

Common (useful) layouts

- Circular layout - nodes positioned in a circle
- Kamada kawai layout - nodes positioned using a path-length cost-function
- Spring layout/Fruchterman-Reingold force directed layout - a physics based algorithm
- Spectral layout - nodes are positioned using the eigenvectors of the Laplacian matrix
- Cose/Cose-Bilkent - implemented only in cytoscape, a physics based spring simulation

Gephi has a more useful library of layouts and produces by far and away the best visualisations but these are best manually implemented from within Gephi due to limited integration with python

Visualising graphs with NetworkX and Holoviews

NetworkX

```
pos = nx.'layout'('graph_object')
draw_networkx('graph_object', 'position_dict')
or
draw_networkx('graph_object', nx.'layout', **kwds)
```

Visualising graphs with NetworkX and Holoviews

Holoviews

```
hv.Graph.from_networkx('graph_object', nx.'layout',
kwargs=dict)
```

Visualising graphs with NetworkX and Holoviews

NetworkX and Holoviews key word arguments (kwds/kwargs)

- arrows, arrowstyle, arrowsize
- with_labels, node_size, node_color, node_shape, cmap
- width, edge_color, edge_cmap, style
- font_size, font_color, font_weight, font_family

NetworkX example

Let's look at a simple example of producing a network visualisation in NetworkX. Open the 'networkx_vis_example.py' file.
The code breaks down as:

- Create some data
- Function to create the graph
- Define the layout
- Define the node and edge attributes: note the number of values for each attribute
- Draw the plot

To draw more complex plots in NetworkX we have to iteratively draw nodes and edges. This means individually pulling in their attributes and plotting them one-by-one. This is not really recommended.

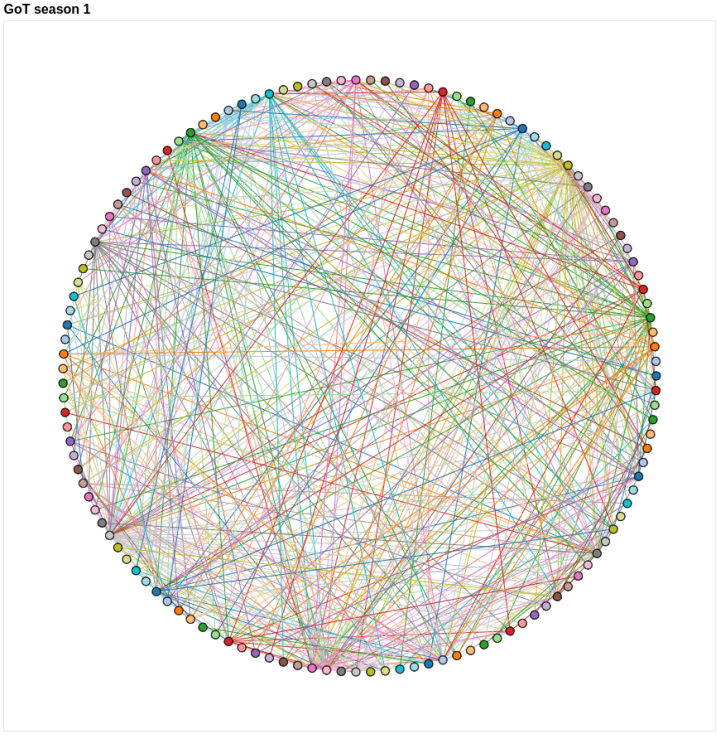
Chord layout in Holoviews

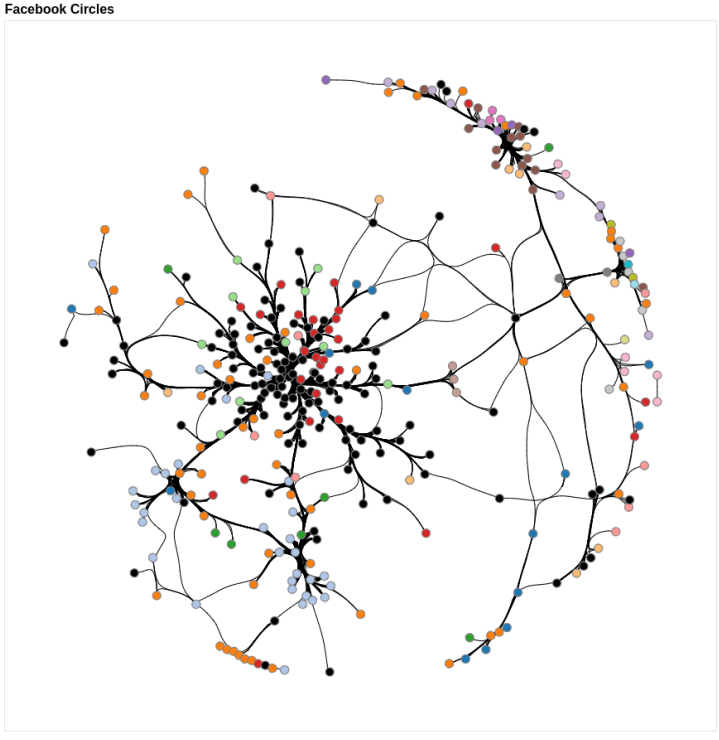
Chord diagrams are a circular layout and can be useful for large graphs

```
chord = hv.Chord((edges, nodes))
chord.opts(
opts.Chord(cmap='', edge_cmap='', edge_color='',
labels='', node_color=''))
```

Holoviews examples

Open the 'basic_holoviews_example.py' file and lets run through this together





Using NetworkX and Holoviews together

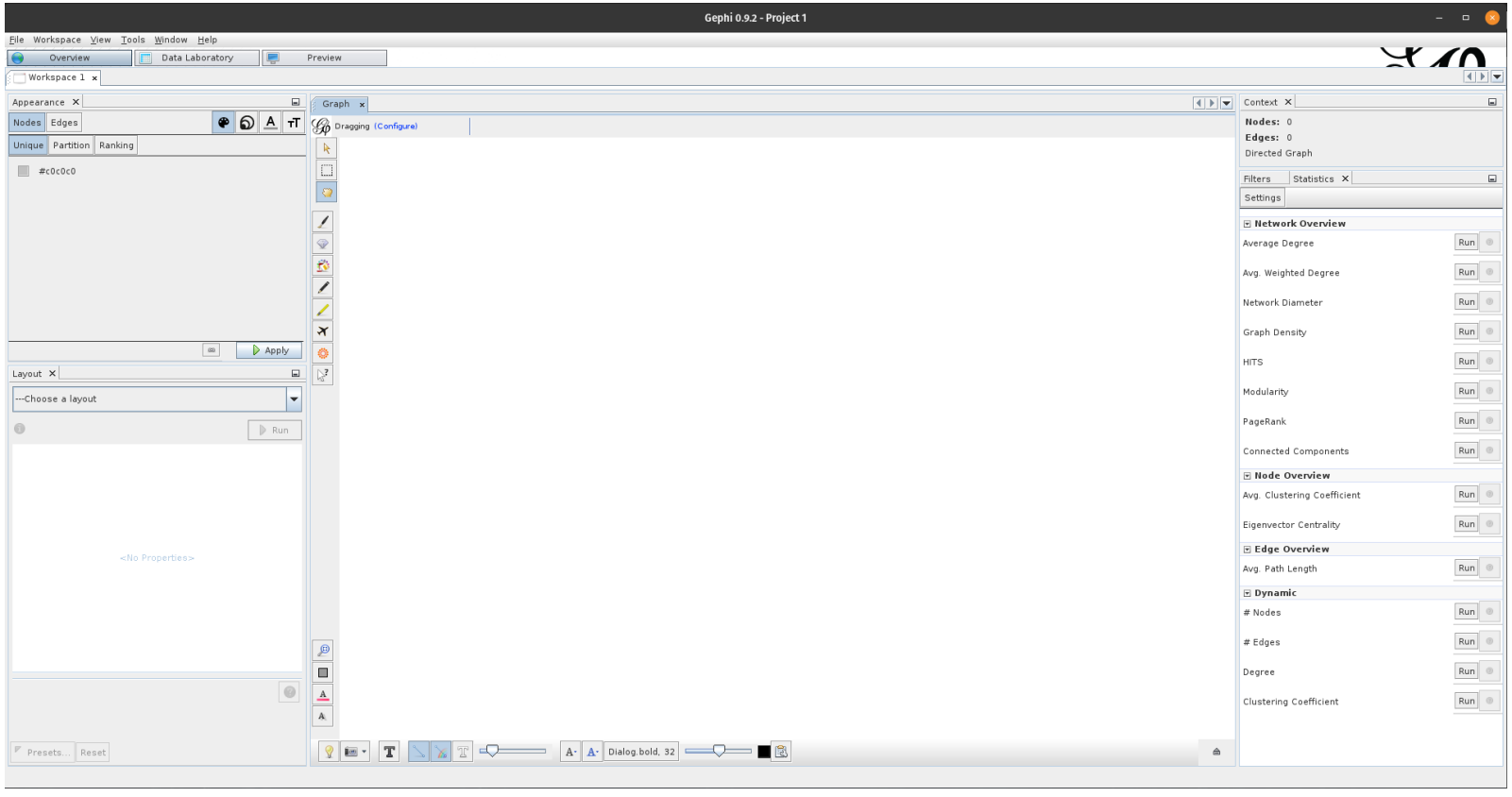
NetworkX is great for analysing our networks and Holoviews is pretty good at visualising them. This is really annoying as we have to create two graphs and use the output of one as the input to another.
Let's have a look at an example of how this can be done.
Open the 'Advanced_holoviews.py' file and we will run through it together.

Visualising graphs with Gephi

- Gephi is a piece of software developed specifically for network visualisation and analysis
- It is the best visualisation engine for producing great looking network graphs
- The built in analysis functionality includes some useful graph metrics
- It is most effectively used when the analysis is conducted with NetworkX and any node and edge metrics are attached to the node and edge lists, exported to CSV and imported into Gephi to produce specific visualisations for presentations and publications

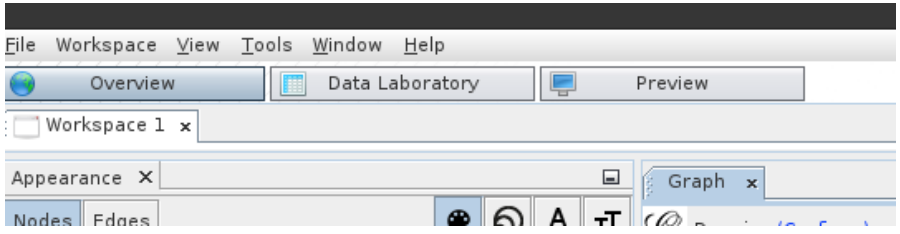
Gephi orientation - Openning screen

Openning screen



Gephi orientation - Main work areas

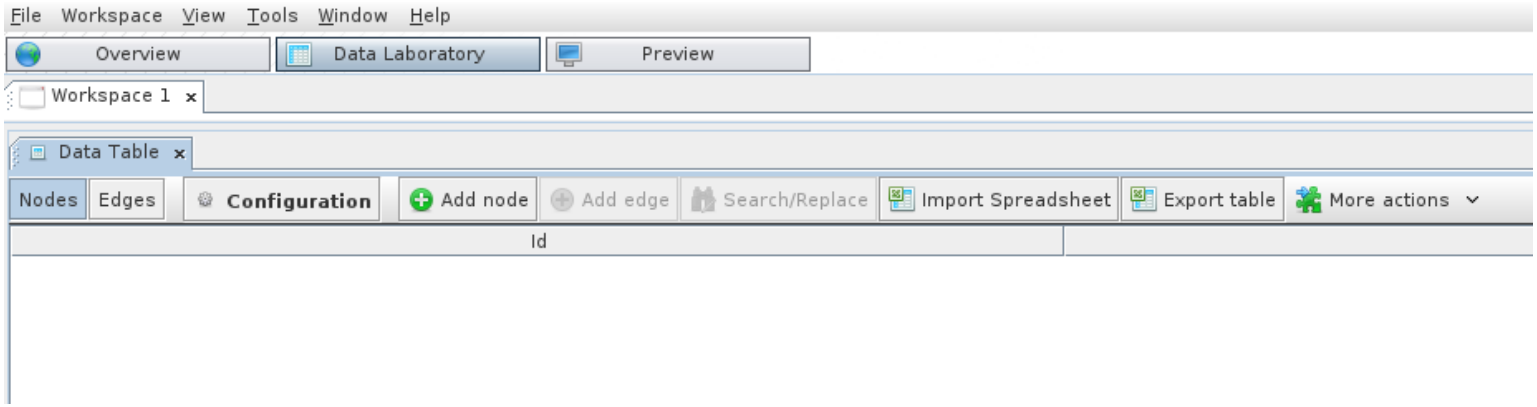
Three main screens: overview, Data Laboratory and Preview



- Overview: Layout, appearance and analysis
- Data Laboratory: Enter or upload data for visualisation
- Preview: refined appearance parameters and image export

Gephi orientation - Data laboratory

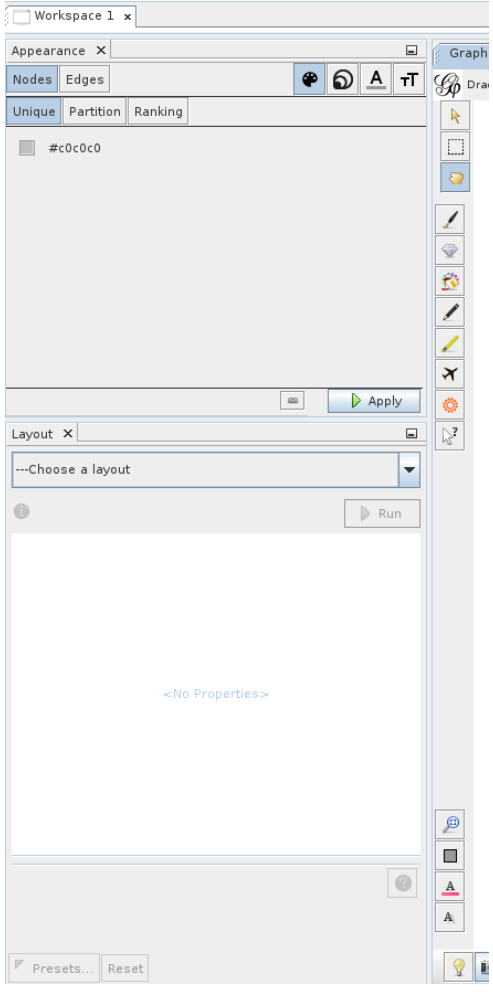
Data laboratory data import



- Separate node and edge data entry
- Enter data manually or import spreadsheet CSV, xls, xlsx
- Nodes: Id and Label columns required
- Edges: Source, Target, Type (directed or undirected) and Id required
- Edge source and target must match node Id

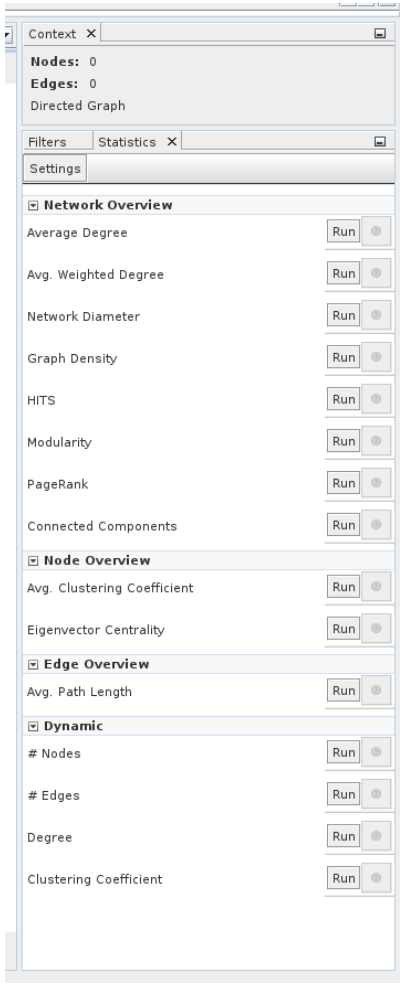
Gephi orientation - Appearance and layout

- Appearance
  - Node and edge formatting
  - Colour, shape, size, text
  - User determined or data determined
- Layout
  - Node position algorithms
  - Each algorithm has its own properties
  - One or more algorithms can be used in combination



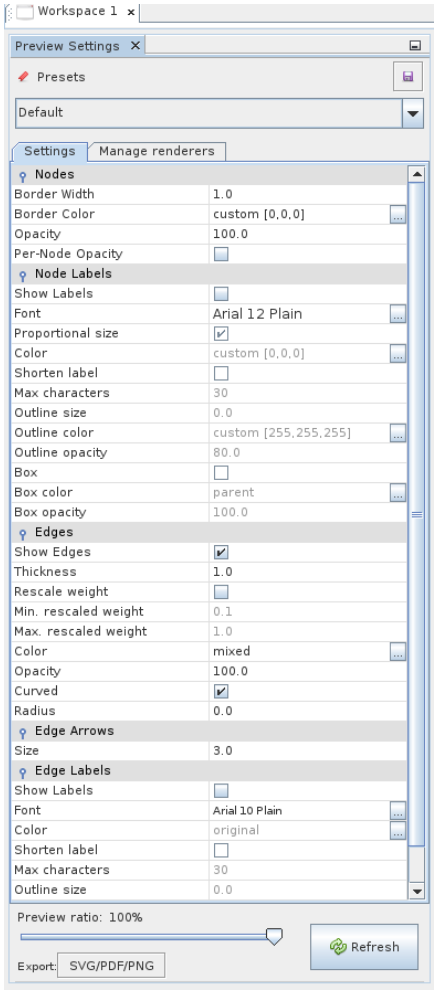
Gephi orientation - Graph metrics

- Network overview
  - Metrics describing the structure of the whole network
  - e.g. average degree, graph density, modularity
- Node overview
  - Average clustering coefficient - grouping across all nodes
  - Eigenvector centrality - relative importance for connectivity
- Edge overview
  - Average path length -



Gephi orientation - Preview

- Detailed visual formatting parameters
- Adjust for nodes, node labels, edges, edge arrows and edge labels
- Takes the input from the overview view and allows refinement
- Need to use the refresh button to update the preview view when changes made to overview
- Export the visualisation view in SVG, PDF and PNG formats
- Presets can be established for repetition



Task time

1. Create a graph of the got_s08 dataset using Gephi
   - Add node labels
   - Colour nodes by modularity class
   - Edge thickness based on weight
   - Determine an appropriate layout
2. Create a graph of the got_s08 dataset using holoviews
   - Adapt the earlier example and use the documentation to see if you can make other changes to the visualisation

Resources

- https://gephi.org/
- https://holoviews.org/user_guide/Network_Graphs.html
- https://holoviews.org/getting_started/index.html
- https://networkx.org/documentation/stable/index.html

Thank you

Thank you for paying attention
Hope you enjoyed the session
Checkout https://www.project-nom.com for more information and training on the use of network-based operational modelling for whole system modelling in healthcare