Analysis of Supply Chain Resilience With Agent-Based Modeling

Ethan Guo
Northwestern University, Evanston, IL
60201
EthanGuo2026@u.northwestern.edu

June 3, 2024

Abstract

Analyzing resilience and determining points of efficiency regarding resilience and costs is vital for businesses to determine optimal logistics configurations. This paper aims to analyze a wide range of system configurations, testing each across simulated environmental stresses with NetLogo Multi-Agent Modeling. Calculated results suggest a correlation of moderate strength between the number of network nodes in a given system and resiliency values adjusted for system output, with a corresponding $p-$value of $2.845 * 10^{-10}$ for linear regression slope, suggesting a statistically significant difference between the slope and 0. However, it should be noted that the data suggest there is no linear correlation between number of network nodes and raw resiliency.

Keywords: Agent-Based Modeling; Supply Chain Resilience; Linear Regression; Data Abstraction

1 Introduction

As retailers and third party logistics becomes more and more commonplace and as globalization continues to increase, the environment in which supply chains and their derivatives find themself becomes increasingly uncertain. As a result, more than ever, it is worthwhile to analyze supply chain resiliency for the purposes of minimizing unnecessary costs while maximizing resilience to external phenomena and stress.

From Wieland, A. and Durach, C.F. (2121) [2], supply chain resilience is defined as a supply chain network's ability to be able to respond and adapt to external phenomena, whether it be inclement weather or road closures. In other words, it is represented both as a system's ability to adapt and transform as well as a system's stability. This study primarily focuses on the latter; the systems modeled do not have the capability to transform, due to their nature of being simple views of supply chain systems.

NetLogo[1] Agent-Based Modeling (ABM) serves as an effective modeling technique because it effectively captures the spatial aspect to supply chain resilience in an object-oriented manner. Mathematical models experience much difficulty in achieving a similar result, and do not carry the same capability in parameter manipulation and experimental data collection that ABM carries.

This study uses ABM to conduct experiments simulating environmental stresses on a basic supply chain model to draw conclusions regarding the key question of how the number of network nodes / configuration of a network can affect the system resiliency.

From S. Chen, K. Tai and Z. Li (2116)[3], the LeCas Modeling Tool, an ABM, is considered one of the most effective methods of simulating a series of interconnected, autonomous agents to produce models that accurately reflect the behavior of a chain system. Although this study does analyze pure supply chain resiliency and utilizes different metrics than those used in S. Chen, K. Tai and Z. Li (2116), ABM was deemed a good fit for the same reasons.

Consequently, this paper is composed in the following manner. Firstly, the methodology and limitations in designing such a model is described in Methodology (II), along with data collection and processing methods. Finally, results will be analyzed and summarized in Results (III), and then interpreted in Conclusions (IV). Citations, including those for data processing methods and software used in both modeling and calculation, can be found at the end.

2 Agent Based Model Experimental Setup

2.1 Agent-Based Modeling

NetLogo Agent-Based Modeling software[1] adopted in the study is an ABM tool that allows for creation of a number of agents to produce models. All 3, used in this study, consist of patches, turtles, and links. Patches are the tiled agents that comprise the "floor" of the model on which turtles, moving agents, can traverse. Finally, links serve as virtual connections between agents to denote special connections that do not apply to the wider agentset of all turtles or all patches. The model runs on a tick time basis, meaning that actions are carried out in discrete steps rather than in continuous time.

2.2 Model Design

By necessity, to model supply chain resilience, two key components must be modeled; the supply chain itself, and the external stress.

Similar to that in S. Huang, S. Sheoran, and H. Keskar(2105)[7] and in S. Chen, K. Tai and Z. Li (2116)[3], a Markov-based model was chosen to demonstrate product flow across network nodes. For the model to be capable of drawing conclusions between different system setups, the net input and output values were to be equal across all experimental trials.

The supply chain model was initially arbitrarily configured to have two tiers of manufacturing centers, one tier each of distribution centers and primary producers, and a variable amount of destination points. However, through heuristic analysis of the model's initial unprocessed outputs as well as through discussion with M. Watson, Undergraduate Professor of Supply Chain Modeling at Northwestern University, this model was determined to be capable of providing results varied enough to draw statistical inference from.

The Markov model has one-way product flow from left to right, utilizing arcs as opposed to edges to simulate this one-way flow. Consequently, by necessity, the model must have a start point and end point. The starting point, primary producers, must generate raw materials for use by the manufacturing centers and serve as the base of the supply chain model. Although traditional supply chain models implement a demand system as a metric to examine the efficiency of a given system, because this study aims to compare a system to itself over a range of external stresses, a demand system was not implemented.

The chosen external stressor for this model is spatially-biased weather, modeled through

diffusion techniques built-in to NetLogo's ABM Software. External stressor modeling is described in greater detail in section 2.4.

2.3 Model Setup and Rules

Firstly, the supply chain model was configured to have five custom defined agent subsets. These consist of primary producers, manufacturing centers, distribution centers, destinations, and deliverers. The first four subsets are stationary turtles for purposes of inventory management and product flow, while the fifth subset, deliverers, are turtles defined for the carrying of products to and from source points and destinations.

Primary producers generate raw materials according to a user-defined parameter (denoted by **raw-regeneration-rate** in Fig. 1) that dictates how many raw materials each producer generates on each tick or timestep. It should be noted that primary producers' net raw material generation remains the same regardless of how many primary producers there are for consistency between systems. Additionally, primary producers are given a field of farm-capacity to limit how many raw materials primary producers can carry at a given time.

Manufacturing centers, the next tier in the supply chain, are given a field for product manufacturing rate, a number randomly generated between 30 and 60. In addition, each manufacturing center is given fields for both available raw materials (input), and available inventory (output of manufactured product). Both of these fields are once again subject to an inventory capacity field. Manufacturing centers comprise two tiers in the final experimental setup as depicted in **Fig. 1**. Finally, manufacturing is also subject to the **manufacturing-conversion-rate**, the ratio of raw materials processed to the number of products created.

Distribution centers do not process products, but instead serve as network nodes that act as an interface between destinations and manufacturing centers. In addition to variables that assign each distribution center a spatial range of supported destinations, distribution centers are assigned an available inventory and inventory capacity field as well to mirror real world warehouse networks.

Destinations serve as metric gatherers by tracking how many deliveries they receive.

Finally, deliverers are the traveling agents that traverse the world to deliver and pick up products from tier to tier. Each deliverer is assigned a home point and a target point from which to pick up and drop off products, respectively. Consequently, each of these

points are assigned with unique values to each deliverer. Each deliverer is also given a carrying capacity, base traveling speed (denoted by **deliverer-speed** in **Fig. 1**), and a direction field for tracking which direction to travel.

The number of each of these agents can be determined by the user, denoted by **num-supply-hubs**(total number of manufacturing centers), **num-tier-0-hubs**(number of tier 0 manufacturing centers), **num-dest**(number of destinations), and **num-dist-centers**(number of distribution centers). However, the number of destinations and primary producers are kept at 30 and $1.5 * num - tier - 0 - hubs$, respectively.

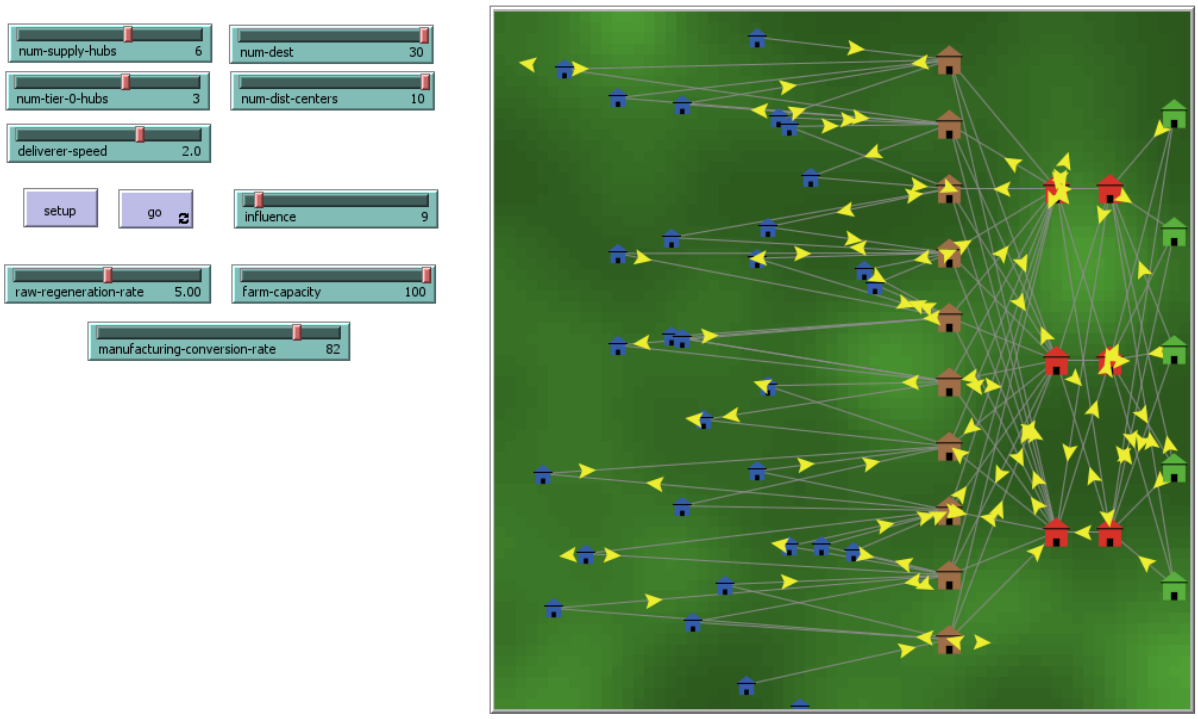

Fig. 1: Initial setup and first 400 ticks of running model with display parameters

2.4 Modeling Environmental Stress

Modeling environmental stress is done on a spatial basis. NetLogo's **diffuse** command serves as a tile-based analog similar to Gaussian diffusion, which creates environmental noise that can be used to model weather[6], and is the primary stressor modeled in this study. Patches, the tiled floor of the world, are assigned a **traversability** value based on the **influence** parameter $(100 - influence)$, and these values are then diffused according to Gaussian diffusion across the world to achieve a simulation of weather's effect on a wide area. In **Fig. 1**, darker green patches represent areas with a lower **traversability**, and vice versa. The **traversability** field affects a number of processes in the supply chain.

Manufacturing centers' manufacturing rate is modified using the **traversability** value

of the patch on which they sit in the following formula.

$$\text{Processing Efficiency} = (1 - \frac{\textbf{influence}}{100})^2 * \frac{\textbf{manufacturing-rate} * \textbf{traversability}}{1.5}$$

Additionally, on each tick, each deliverer's forwards travel distance is determined by the traversability value of the patch they are currently on according to the following formula.

$$\text{Speed} = \frac{\textbf{deliverer-speed} * \textbf{traversability}}{1.5}$$

2.5 Experimental Parameter Selection

Initial constants and parameters, including **num-dest**, 1.5 in the above formulas, and **deliverer-speed** were chosen through heuristic analysis. With the final experimental setup, parameters were changed until the best-performing model, with 10 distribution centers and 5 manufacturing centers of each tier, could perform at capacity without inventory bottlenecks at any location. With this setup, the ceiling is removed for performance (assuming that introduction of influence can only negatively impact system performance). However, it proved to be infeasible to remove the performance floor through heuristic analysis, and concessions were made to accuracy described in section 3.

The NetLogo model contains plots for monitoring inventory bottlenecks for diagnostic purposes such as the aforementioned process of determining equation constants and set parameters. An example of the final experimental configuration's resultant graphs can be seen in Fig. 2, whereupon each agent's inventories reached an equilibrium steady state between their inventory floors and ceilings, effectively functioning at capacity.

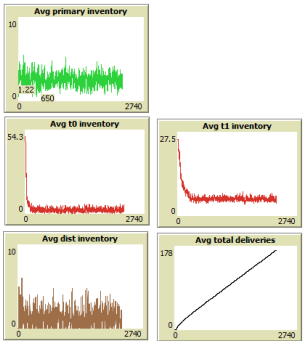

Fig. 2: Diagnostic graphs

3 Data Collection Methodology

3.1 BehaviorSpace

BehaviorSpace is an experimental data collection tool built into NetLogo that allows users to simulate experiments over varied parameter values with specified startup conditions, until an end condition is met. For the purposes of this experiment, BehaviorSpace was used to vary over influence values, ranging from 0 to 100. Due to the computational limitations, values were varied in steps of 5 from 0 to 100 to preserve a high degree of information while greatly reducing the dimensionality of data that would be produced, allowing for easier data processing and storage.

To arrive at different system configurations, the number of manufacturing and distribution centers were varied, while maintaining the number of destinations as well as net input of raw materials as described in section 2.1. This way, although the "hidden layers" of the supply chain are varied, the net input and output conditions remain the same for all systems to allow for output comparison between different systems. BehaviorSpace was run at 30 iterations for each value of influence, resulting in 630 total runs for each system configuration, which were determined below.

3.2 System Configuration Selection

Due to the nature of the model, the number of potential BehaviorSpace iterations is determined by the following formula, accounting for all possible configurations of manufacturing and distribution centers, as well as influence values.

$$10 * \sum_{n=1}^{9} \binom{n}{1} * 630 = 6300 * \sum_{n=1}^{9} \frac{n!}{(n-1)!} = 283500 \tag{1}$$

To pare down the number of data points, an algorithm was designed in Python to deterministically generate configurations starting with a preset number of destination centers which were ranged from 1 to 10. From this algorithm, the data were processed via Principal Component Analysis to reduce the 450 configurations to 50, retaining roughly 90% information content.

Each of these configurations was then passed into BehaviorSpace for manual data collection. Data were then written into CSV files by NetLogo.

3.3 Data Collection

The primary metric gathered in BehaviorSpace trials was the average fulfillment to each destination per 100 ticks, or timesteps. This value is re-calculated every 100 ticks, to generate both a discrete graph and a running graph to track the value. Because the running graph matched the discrete graph to a high degree as shown in Fig. 3, the running graph's value was chosen as the metric gathered due to ease of collection.

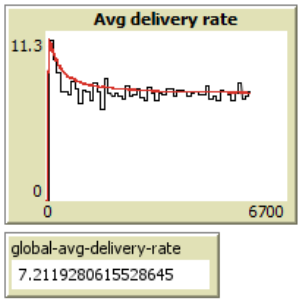

Fig 3: Average Delivery Rate graph

The discrete graph, penned in black, better highlights small scale swings in delivery rates and smaller scale trends, while the global average, penned in red, averages across all time passed to produce a more consistent value and smoother graph that reflects the aggregate of all discrete changes.

Across the BehaviorSpace trials, for each given system, 30 delivery rates were calculated for each influence level from 0-100 in steps of 5, generating 630 data points for each system. To further collapse this data, the average for each influence level was taken to give an array of 21 average delivery rates for each system configuration. The purpose of taking the average is to generate a graph that describes how the system's performance changes as you increase the external stress.

Through initial testing as well, it was discovered that given the random nature of Gaussian Diffusion in model setup, outliers can result despite the controlled experimental environment. To counter this, averages were taken to decrease the effects of these outliers.

3.4 Data Processing

Data Processing was primarily done in Python. CSV outputs from BehaviorSpace were read to process data, assign scores, and perform all data processing calculations.

Because each system has inherent differences in their fulfillment averages even without influence, the data for each array of 21 delivery rates was normalized with respect to the average delivery rate measured without any external influence. The normalization allows for resultant measurements to represent the performance of the system strictly as a dependent variable of influence, independent of their baseline performance.

Using these 21 delivery rates, a graph can be generated plotting relative average delivery rate over influence. Some examples are shown in Fig. 4 below.

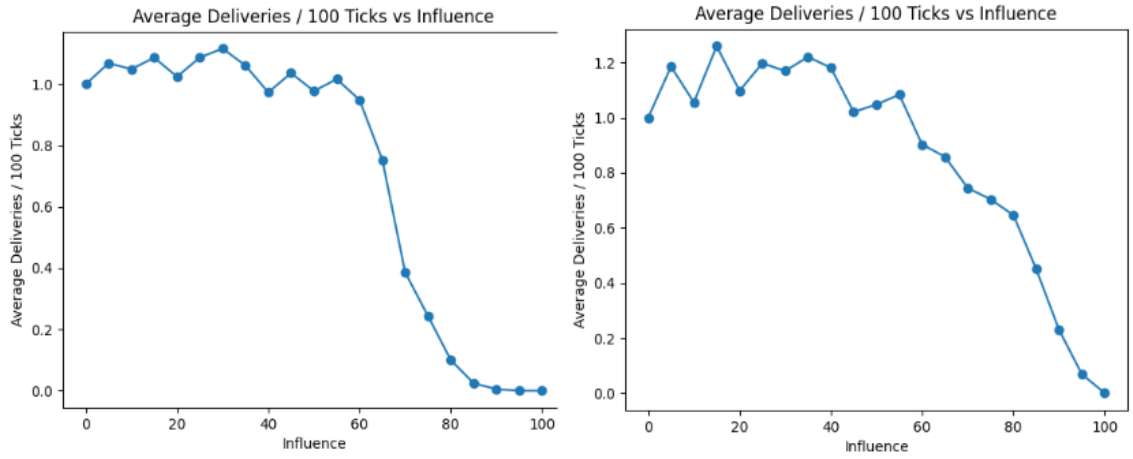

Fig. 4: L: Resilience Graph for 8 $t1$, 2 $t0$, and 4 $dist$. R: Resilience Graph for 5 $t1$, 5 $t0$, and 10 $dist$. Generated via matplotlib.pyplot

Qualitative analysis of the two graphs can yield important information. In Fig. 4, the leftmost resilience graph demonstrates a steep decline in Delivery Rate between $influence \in [50, 80]$ but quantitative analysis is required to generate data summaries. Consequently, two initial metrics were chosen to abstract these graphs into "resilience factors".

- **Metric 1: Resilience Score**
  The area under the curve is traditionally given by $\int_0^{100} f(x)dx$. However, due to the discrete nature of the data, a trapezoidal approximation must be used, given by:

$$A(f(x)) = \int_0^{100} f(x)dx \approx \sum_{k=1}^{21} \frac{f(x_{k-1}) + f(x_k)}{2} * 5 \tag{2}$$

  where for each normalized array of 21 points $f(x)$, where x_1 denotes the first element of the arrays. Note that 5 is substituted for the base of each trapezoid because of the nature of the construction of the data arrays.

  This provides a metric that can be used to compare and rank each system with all other systems, allowing for macro analysis of all systems.

- **Metric 2: Fréchet Distance**
  Fréchet distance is defined to be the infimum of the maximum over all reparameterizations of functions A and B. In mathematical notation, Fréchet distance is

defined as

$$F(A, B) = \inf_{\alpha,\beta} \max_{t\in[0,1]} \{d(A(\alpha(t)), B(\beta(t)))\} \tag{3}$$

where d is a function of distance, and α and β are reparameterizations.

For this study's purposes, Fréchet distance was adapted for discrete data sets, and was represented in python as the greatest difference between points in datasets corresponding to the same x value, influence. In mathematical notation, the discrete Fréchet distance, as shown in T. Wylie and B. Zhu (2014)[8], is represented as

$$F_d(A, B) = \inf_{\alpha:[1:m+n]\to[0:m],\beta:[1:m+n]\to[0:n]} \max_{t\in[1:m+n]} \{d(A(\alpha(t)), B(\beta(t)))\} \tag{4}$$

where m, n are the number of points in each array, or 21, in this case.

For the purposes of comparing resiliency graphs, the distance function chosen will be the L2 Norm, or Euclidean Distance. This provides a metric that can be used to compare two systems, but no more. Although a Fréchet adjacency matrix could be generated, it was deemed better suited for direct comparisons as opposed to a metric to rank graphs by.

During data collection, however, it was discovered that very low-performing systems, that is systems with a relatively low performance in terms of non-normalized fulfillment values, experienced almost no difference with or without stress, a sort of cushioning effect. Through further analysis and consultation with experts, the reason was determined to be because there was no room for the system to perform worse. As a result, a new metric was created to account for this effect.

- **Metric 3: Weighted Resilience Score**
  After system configurations were first tested for the best performing and lowest performing systems, it was found that the averaged average fulfillment rates would fall between values of 0.3 at the lowest and 9 at the highest. Through qualitative analysis, it was determined that the cushioning effect was roughly exponential, increasing rapidly as the performance decreased. Thus, to penalize the lower performing systems, the following formula was used to calculate a weighted resilience score with natural log as a penalty.

$$W(f(x)) = \sum_{k=1}^{21} \frac{f(x_{k-1}) + f(x_k)}{2} * 5 + k * \ln(x_1)_0 \tag{5}$$

where $f(x)$ is the array of points, k is some penalty weight, and $(x_1)_0$ is the non-normalized first value of each array $f(x)$.

To determine a value for k, a first baseline was calculated to find penalties. Penalties were determined to range from 30 for the worst performing systems to 0 at the highest end. Combined with the averaged average fulfillment rates above, the following calculations were done to determine k.

$$P_{min} = -k * \ln 0.3 \implies 30 = -k * \ln 0.3 \implies k \approx 25$$

Which, by plugging in to **(4)**, gives the following final formula.

$$W(f(x)) = \sum_{k=1}^{21} \frac{f(x_{k-1}) + f(x_k)}{2} * 5 + 25 * \ln (x_1)_0 \tag{6}$$

This provides a metric that can be used to compare and rank each system to all other systems taking into account both the cushioning effect as well as system performance, a practical measure of system performance as a function of weighted resilience.

Data was collected by running BehaviorSpace trials for each configuration, outputting 30 trial run average fulfillment rates for each influence level per configuration, resulting in 31500 data points.

Data points were averaged across influence values for each configuration, compressing the data down to fifty 21 point arrays. Metrics **1** and **3** were gathered using **(2)** and **(6)** implementations in python utilizing numpy and csv packages, and graphs were generated via matplotlib.pyplot.

4 Results and Analysis

In results summary, Total Network Nodes is defined as the sum of the number of distribution centers and manufacturing centers as the number of destinations remains the same, and the number of primary producers varies despite having the same net output. Therefore, there is a maximum of 21 and minimum of 3.

4.1 Total Network Node Analysis

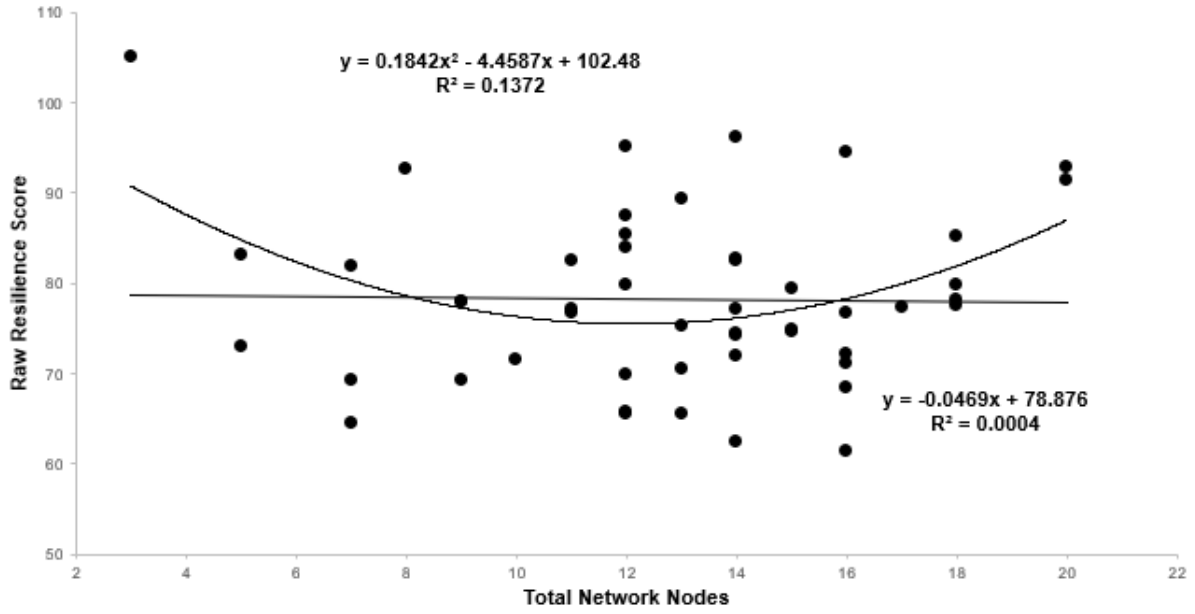

Fig. 5: Scatterplot plotting raw resilience scores against number of network nodes.

From initial analysis, there appears to be no correlation betwen the number of total network nodes and the raw resilience score, which measure the system's resilience as a function of influence relative to its baseline performance. Using R[4] and visualization tools[5] to conduct a statistical analysis of the data, the data suggest that there is no linear correlation between the number of total network nodes and raw resilience score.

Additionally, testing for goodness of fit of the slope of the line of best fit denoted by the equation in the bottom left yields a p-value of 0.8967. Because $p = 0.8967$ is greater than $\alpha = 0.05$, we fail to reject H_0. Thus, the data suggest that the slope of the line of best fit is 0.

Conducting polynomial regression tests, however, yields more interesting results. With a R^2 of 0.1372, the data suggest a weak correlation between the number of total network

nodes and the raw resilience scores, given by the following equation.

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 = 102.48 - 4.4587x + 0.1842x^2 \tag{7}$$

Analyzing goodness of fit for β_1 and β_2 gives p-values of $p_1 = 0.0096$ and $p_2 = 0.0089$, respectively. Because $p_1, p_2 < \alpha = 0.05$, we reject H_0 in both cases. Therefore, the data suggest that both β_1 and β_2 are not 0.

One potential explanation for the shape of **Fig. 5** is the cushioning effect aforementioned in section 3.4. The cushioning effect would result in a higher resilience score for lower performing systems, while higher performing systems display a higher resilience by nature, as they do not experience the cushioning effect. Consequently, it would be expected for the graph to exhibit two maxima, one at both ends, as described by **(7)**.

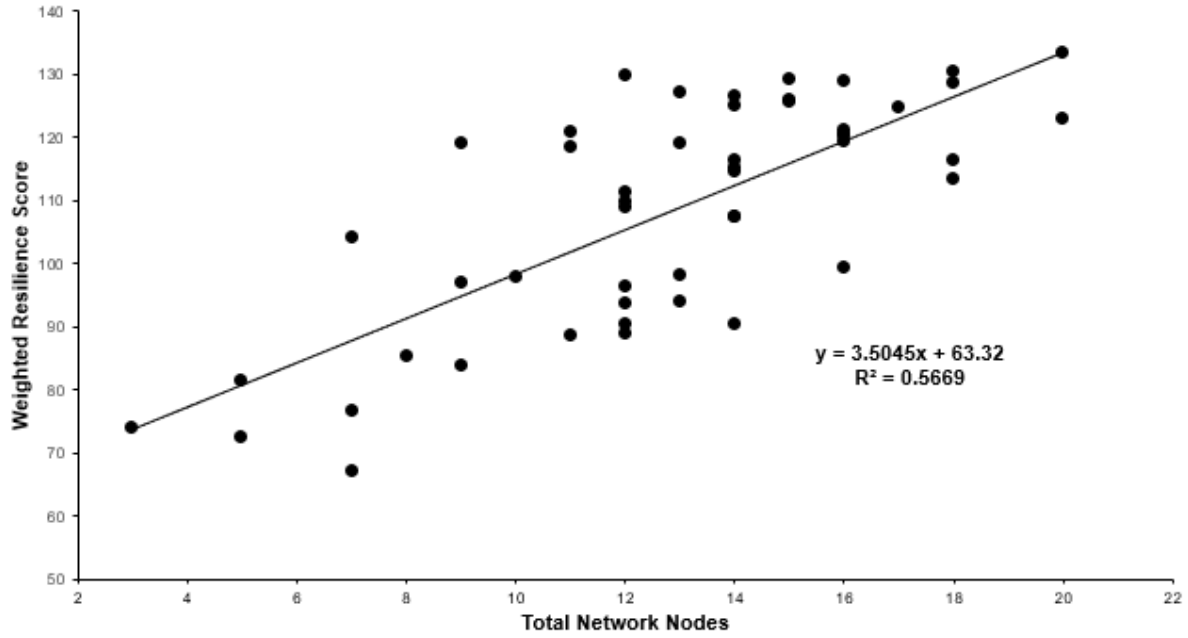

Fig. 6: Scatterplot plotting weighted resilience scores against number of network nodes.

A much clearer trend can be observed when plotting weighted resilience scores instead of raw resilience scores as seen in **Fig. 6**. With an R^2 of 0.5669, there is a moderate correlation between the number of total network nodes and weighted resilience scores where 56.7% of the variance in weighted resilience score is explained by the number of network nodes. Conducting linear regression analysis for the goodness of fit of the slope= 3.5045 gives a p-value of $2.845 * 10^{-10} < \alpha = 0.05$ Therefore, we reject H_0 and the data suggest that the true slope is not 0.

By the nature of its construction, weighted resilience score considers the baseline performance of each network by default. However, for the best performing models who

achieved a baseline performance of an averaged 9 deliveries a tick, the penalty function in **(6)** actually gives a performance bump of up to 50, biasing systems that perform better as well as penalizing systems that perform worse. As a result, these results should be interpreted as a measure of practicality or system efficiency taking into account resilience, as opposed to strictly a measurement of pure resilience.

From **Fig. 5** and **Fig. 6**, systems with the highest and lowest number of network nodes exhibit the greatest degree of resilience in a polynomial relationship of weak strength. However, taking into account practicality and overall system performance with the same net input and output conditions, higher network node count systems performed best both in terms of net output as well as resilience in a linear relationship of moderate strength.

4.2 Individual Metric Analysis

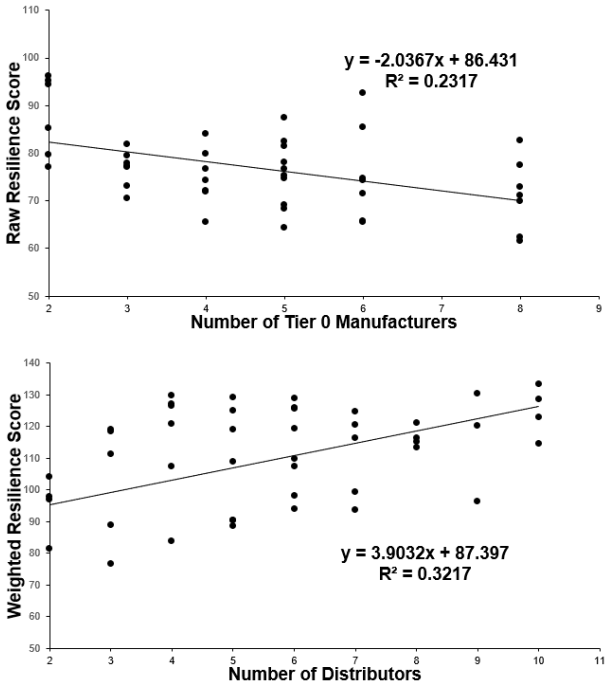

Fig. 7: U: Scatterplot plotting raw resiliency score against number of tier 0 manufacturing centers L: Scatterplot plotting weighted resiliency scores against number

A number of metrics were plotted against raw resilience score to identify any potential correlations. These metrics included the number of tier 0 and tier 1 manufacturers, the number of distribution centers, and the ratio between tier 0 and tier 1 manufacturers.

Interestingly, the metric with the greatest correlation with raw resilience scores was the number of tier 0 manufacturers, and the metric with the greatest correlation with weighted resilience scores was the number of distributors. Plotting these together yielded moderately weak correlations of $R^2 = 0.2317$ and $R^2 = 0.3217$. Through qualitative analysis, however, the graphs appear to peak at $\#t0 \approx 6$ and $dist \approx 5$ rather than follow a strictly linear relationship.

Further linear regression analysis for slope yields p-values of 0.0004017 and 0.0000175 for the upper and lower graphs, respectively. With $p = 0.0004017, 0.0000175 < \alpha = 0.05$, we reject H_0 in both cases. Thus, the data suggest that the slopes are not 0.

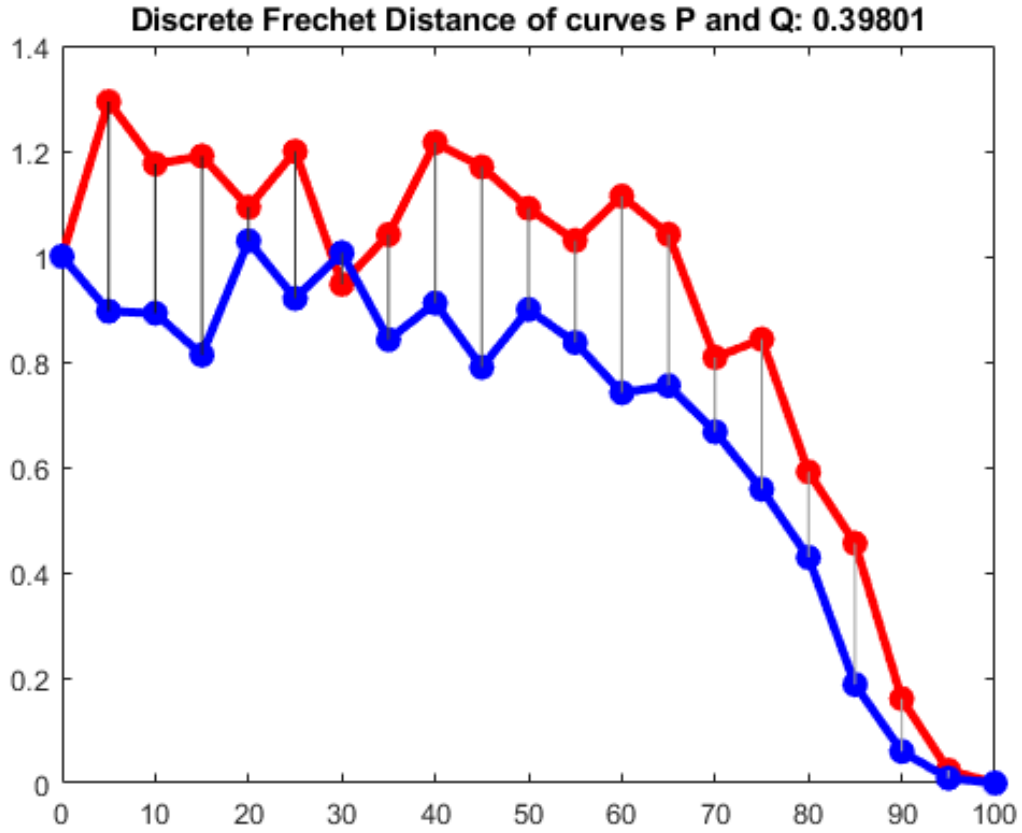

Fig. 8: Depiction of Fréchet distance between second highest and lowest raw resiliency score graphs generated via MATLAB

The lowest performing configuration in terms of raw resilience score consisted of 7 distribution center, 8 tier 0 manufacturing centers, and 1 tier 1 manufacturing center. The highest performing configuration consisted of 1 each of distribution centers, tier 0, and tier 1 manufacturing centers. Between all graphs, the largest Fréchet distance was found to be between the second highest performing $(2t0, 2t1, 10dist)$ and lowest performing resilience score system configurations.

Fréchet analysis was done in MATLAB using Zachary Danziger (2014)'s file exchange[9]. From **Fig. 8**, Fréchet distances tended to be larger towards the first ≈ 65 influence values, including the largest Fréchet distance measured at $influence = 45$. Towards the primary slope drop off towards the end of the x-axis, we observe that the Fréchet distances maintain a lower value, suggesting that independent of system configuration, a similar drop off can be expected.

References

[1] Wilensky, U. (1999). NetLogo. http://ccl.northwestern.edu/netlogo/. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.

[2] Wieland, A. and Durach, C.F. (2021), Two perspectives on supply chain resilience. J Bus Logist, 42: 315-322. https://doi-org.turing.library.northwestern.edu/10.1111/jbl.12271

[3] S. Chen, K. Tai and Z. Li, "Evaluation of supply chain resilience enhancement with multi-tier supplier selection policy using agent-based modeling," 2016 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), Bali, Indonesia, 2016, pp. 124-128, doi: 10.1109/IEEM.2016.7797849.

[4] Posit team (2024). RStudio: Integrated Development Environment for R. Posit Software, PBC, Boston, MA. URL http://www.posit.co/.

[5] Statistics Kingdom.(2017).Multiple Linear Regression Calculator. (May 30, 2024)[web application]. https://www.statskingdom.com/410multilinearregression.html

[6] Lizao Li et al. ,Generative emulation of weather forecast ensembles with diffusion models.Sci. Adv.10,eadk4489(2024).DOI:10.1126/sciadv.adk4489

[7] Samuel H. Huang, Sunil K. Sheoran, Harshal Keskar, Computer-assisted supply chain configuration based on supply chain operations reference (SCOR) model, Computers & Industrial Engineering, Volume 48, Issue 2, 2005, Pages 377-394, ISSN 0360-8352, https://doi.org/10.1016/j.cie.2005.01.001.

[8] Tim Wylie, Binhai Zhu, Following a curve with the discrete Fréchet distance, Theoretical Computer Science, Volume 556, 2014, Pages 34-44, ISSN 0304-3975, https://doi.org/10.1016/j.tcs.2014.06.026.

[9] Zachary Danziger (2024). Discrete Frechet Distance (https://www.mathworks.com/matlabcentral/fileexchange/31922-discrete-frechet-distance), MATLAB Central File Exchange.