

Article

A Mathematica-Based Interface for the Exploration of Inter- and Intra-Regional Financial Flows

Kyriaki Tsilika ^{1,2} 

¹ School of Economics and Business, Department of Economics, University of Thessaly, 38333 Volos, Greece; ktsilika@uth.gr; Tel.: +30-2421074367

² School of Social Sciences, Department of Business Administration, Hellenic Open University, 26335 Patras, Greece

Abstract: This work surveys the use of directed weighted graphs in conducting comparative static analyses. The paper discusses the implementation of a computer-aided process for building spreadsheet-based graph models for inter- and intra-regional financial flows. The graph-theoretic techniques are programmed to enable the interactive visualization and analysis of financial data using Wolfram technologies (i.e., Mathematica software v. 11.3 or later, Wolfram player v. 12 or later). The paper describes the workflow for several interactive visualizations applicable to financial networks. The author provides four programs, written in the Wolfram language, that customize input–output financial models by combining the Manipulate command with built-in Mathematica functions and functions of the IGraph package (IGraph/M). The study is a tutorial article for the generation of a suite of visual schemes that provide patterns, practices, and roadmaps of the financial markets across the globe.

Keywords: graph modeling; functional programming; Mathematica software; Wolfram language

MSC: 05C20; 05C21; 05C51; 68R10



Citation: Tsilika, K. A

Mathematica-Based Interface for the Exploration of Inter- and Intra-Regional Financial Flows.

Mathematics **2024**, *12*, 877.

<https://doi.org/10.3390/math12060877>

Academic Editor: Andrea Scozzari

Received: 23 January 2024

Revised: 13 March 2024

Accepted: 14 March 2024

Published: 16 March 2024



Copyright: © 2024 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Most financial infrastructures collect and manage data in a file format supported in MS Excel. The framework for building spreadsheet-based, graph-assisted financial models is presented in the sections that follow. The framework guides the step-by-step, efficient development of effective models that can support financial decision-making and provide useful insights to market decision-makers, enterprises, and investors.

All the modeling aspects proposed here are demonstrated using real-world data from [1] on green bond allocation proceeds. In the computational paradigms, the flows of the transaction proceeds are depicted using a graph-aided representation method. Each node represents a single counterparty in the market. Each edge represents the flow of green bond proceeds.

In business models, it is often useful to have interactive graph representations, with the chart input range responding to any changes in the graph simultaneously. This approach permits specific graphs to be displayed on request [2]. In this direction, we use the computer algebra system Wolfram Mathematica (v. 11.3 or later) to display practices, patterns, and roadmaps of the intercontinental green bond market with interactive, graph-based visualizations. Graph theory and model-building tools are developed in tandem with techniques for solving practical problems by using the symbolic program Mathematica [2–6]. Mathematica-based programming integrated into graph modeling and analysis is in the sphere of interest from 1990 to 2022, as seen from the existing relevant literature. Financial networks in Mathematica, though, have not been studied extensively, and this research gap will fill the present paper.

Mathematica provides various visualization modules to create tables, heatmaps, maps, and traditional statistical charts, with a variety of customizations with controls added for user intervention. Users can create interactive visualizations that express the desired output using filters, drop-down lists, and so on. Some functionalities rely on the graph design built-in functions of the Wolfram language (<https://reference.wolfram.com/language/guide/GraphsAndNetworks.html>, accessed on 15 January 2024). Other functionalities link to the *igraph* package for Mathematica, *IGraph/M* (v. 0.6.5) [7].

This paper is a tutorial for building spreadsheet-based graph models to explore inter- and intra-regional financial flows. Reproducibility is possible for people who are familiar with the basics of using the Wolfram language including graph theory and network analysis functions. Graph modeling with Mathematica provides a comprehensive overview of the interacting agents and the strategic formation of links. Input–output financial flows are interpreted into visual elements.

Financial network modeling is proposed in several articles via powerful and versatile tools for financial analysis, yet it poses some challenges and limitations. Open-source languages that provide the visualization and structural analysis of relations and dependencies require programming knowledge of an advanced level. Wolfram language is a symbolic language, deliberately designed for researchers who understand mathematics. It quickly expresses complex ideas in computational form, in a mathematics-oriented internal logic. While each programming language has its own use case or specific fields, Wolfram language differs because it is knowledge based. This means that it already knows a lot—so the user has to tell it much less to get it to perform the required tasks. Wolfram language assumes no prior knowledge of programming, while coding elsewhere could be troublesome for non-computer scientists [8].

Widely known no-code software that visualize results in network analysis do not go as deep in graph analytics as a researcher would desire (i.e., develop algorithms to identify the systemic importance of nodes in a network).

Above and beyond everything else, the key advantage of the proposed approach via the use of Wolfram Mathematica is the creation of interactive outputs. Interactivity in outputs is enabled with the *Manipulate* function of Wolfram language and permits so-called scenario planning. In financial analysis, envisioning different possible future scenarios and assessing their potential impacts on an organization or system is a meaningful skill for financial analysts.

Interactivity in network visualization serves a wide variety of purposes and makes present networks more powerful than ever before. Interaction, for example, provides spatial and temporal context, allows for particularization and increased detail, affords scenario planning, and facilitates the exploration of market behavior.

The paper is structured as follows: The next section briefly reviews graph modeling in Mathematica. The required data structure and data loading are given in detail. Section 3 presents an informal description of the interactive visualization of financial networks in a collection of ordered steps. Also, it includes four precise step-by-step plans that form four interactive illustrations to explore financial data with controls to create the desired output. While in Section 3, computer codes run into a Mathematica notebook, Section 4 suggests the development and customization of computer models in computable document format, proposing a visualization application for financial networks. Section 5 summarizes the results of this study. Section 6 discusses the benefits of the proposed interface and gives some examples of the application of the interface in real-world financial analysis. Section 7 contains technical guidelines. The last section concludes the paper.

2. General Specifications

2.1. Graph-Assisted Models for Financial Flows

In the sections that follow, the network of relationships between market participants is set from information derived from the real-world data of [1]. In this financial network, nodes represent regions as counterparties, while edges represent the flow of green bond

proceed allocations between pairs. An edge between two nodes exists if there is at least one transaction between the two counterparties. For each connected pair of nodes i and j , the edge is weighted with the gross mark to the market value of all transactions between the two nodes/counterparties. The arrows that represent the edges are directed from the allocator to the recipient region.

The network in Figure 1, which represents financial transactions, is a weighted, directed, and aggregate network, as are the rest of the financial networks present in this work. Aggregate networks are obtained by summing the weights of multiple edges between two nodes. The weight of an edge is the total sum of the weights across all the years of the period 2006–2017, i.e., the weight is the sum of the absolute market values of all transactions between two nodes within the period 2006–2017. Various metrics computed on multi-layer networks and aggregate networks have elaborated structural analyses of financial markets [9–16]. The complex network modeling of systems formed by financial agents has been proposed in [17–19].

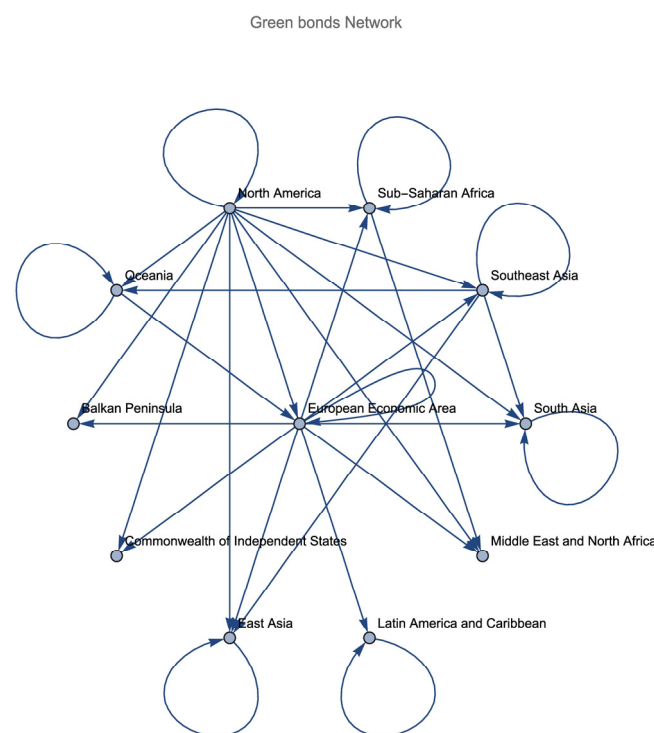


Figure 1. The simulated region-to-region transfer network as created in Mathematica. Edges represent aggregated allocations between regions over the 2006–2017 period.

To increase the level of detail, such as the green bond transfer network of a specific region (or for a specific allocation year), additional controls can be added to the basic view of Figure 1. Then, we can filter the view to focus on a specific geographical region (or allocation year) (see Section 3).

2.2. Identification of Inputs and Outputs

The prerequisites for building graph-based financial models are given below. The outputs are the visualizations needed to achieve the objective of the task.

The inputs are data and information for the implementation of the required outputs.

Information needed

- The counterparties of the financial market (can be financial institutions, countries, regions, etc.).

Inputs

- The adjacency matrix for every year of the study period that represents all the possible relationships between the counterparties. The structure of the adjacency matrix is presented in Table 1. The adjacency matrix has the dimensions $n \times n$.

Table 1. The adjacency matrix structure.

Source/Destination	Destination 1	Destination 2	...	Destination n
source 1 *	a_{11}	a_{12}	...	a_{1n}
source 2	a_{21}	a_{22}	...	a_{2n}
...	...	...	...	...
source n	a_{n1}	a_{n2}		a_{nn}

* Source i denotes each financial allocator. Destination j denotes each financial recipient. The entries a_{ij} obey the following rule: $a_{ij} = \begin{cases} 1, & \text{if counterparties } i \text{ and } j \text{ interact} \\ 0, & \text{if counterparties } i \text{ and } j \text{ do not interact} \end{cases}$

- The weighted adjacency matrix for every year of the study period (a weight w_{ij} is defined by the total number of transactions between the two counterparties). The structure of the adjacency matrix is presented in Table 2. The weighted adjacency matrix has the dimensions $n \times n$. If a source does not connect to a destination, the dedicated transaction is equal to 0.

Table 2. The weighted adjacency matrix structure.

Source/Destination	Destination 1	Destination 2	...	Destination n
source 1 *	w_{11}	w_{12}	...	w_{1n}
source 2	w_{21}	w_{22}	...	w_{2n}
...	...	...	...	...
source n	w_{n1}	w_{n2}		w_{nn}

* Source i denotes each financial allocator. Destination j denotes each financial recipient. w_{ij} is the total number of transactions between the two counterparties.

The Excel (2013 or later, Microsoft 365) spreadsheets containing the adjacency and the weighted adjacency table are imported in Mathematica with the Import function.

To use the adjacency and the weighted adjacency tables in the graph construction and representation functions, the process involves using the Take function to extract the numerical submatrix of the adjacency and the weighted adjacency matrix (excluding the headings in the first row and column). Next, infinity symbols substitute zeros, as follows:

$$\begin{aligned} r1 &:= \text{Take}[r, 2 ;; 12, 2 ;; 12] \\ r11 &:= r1 /. \{0. -> \text{Infinity}, 0 -> \text{Infinity}\} \end{aligned}$$

where r is the weighted adjacency matrix (as defined in Table 2) given in Mathematica's matrix arrangement as a list of lists.

Outputs

- Interactive visualization of the graph model representing the network of financial relations with marking graph segments;
- Automatic creation of subgraphs to focus exploration;
- Interactive visualization of the adjacency matrix representing the network of financial relations;
- Dashboards to control the view of financial flows in two or more different illustrations.

3. Design Process and Task Analysis

Each one of the programming modules is developed in Wolfram language with a step-by-step plan for a computational procedure that begins with an input value and yields an output value in a finite number of steps.

Interactivity is achieved using a single command, *Manipulate* (<https://www.wolfram.com/wolfram-u/courses/visualization-graphics/understanding-manipulate-vis021/>, accessed on 15 January 2024), which generates an interactive object containing one or more controls that can be used to vary the value of the parameter within the parameter range. The output of *Manipulate* is very much like a small widget: it is not just a static result; it is a running program that can be interacted with. Since Wolfram language provides various customizations, the level of analysis can be increased with controls for time filtering and geographical specifications, combining the *Manipulate* command with built-in Mathematica functions and functions of the IGraph package (IGraph/M).

The interactive part of the *Manipulate* command is set by control elements (i.e., drop-down lists) for either of the following:

- The geographical region (here, the categorical variable “region” includes 11 regions);
- Year of study (here, the categorical variable “allocation year” includes the years of the 2006–2017 period).

Below, we describe the evaluation part of the *Manipulate* command that generates a customized graph that highlights the connections of selected nodes and is depicted in Figure 2a–c. Each visual scheme highlights the connections of the selected region.

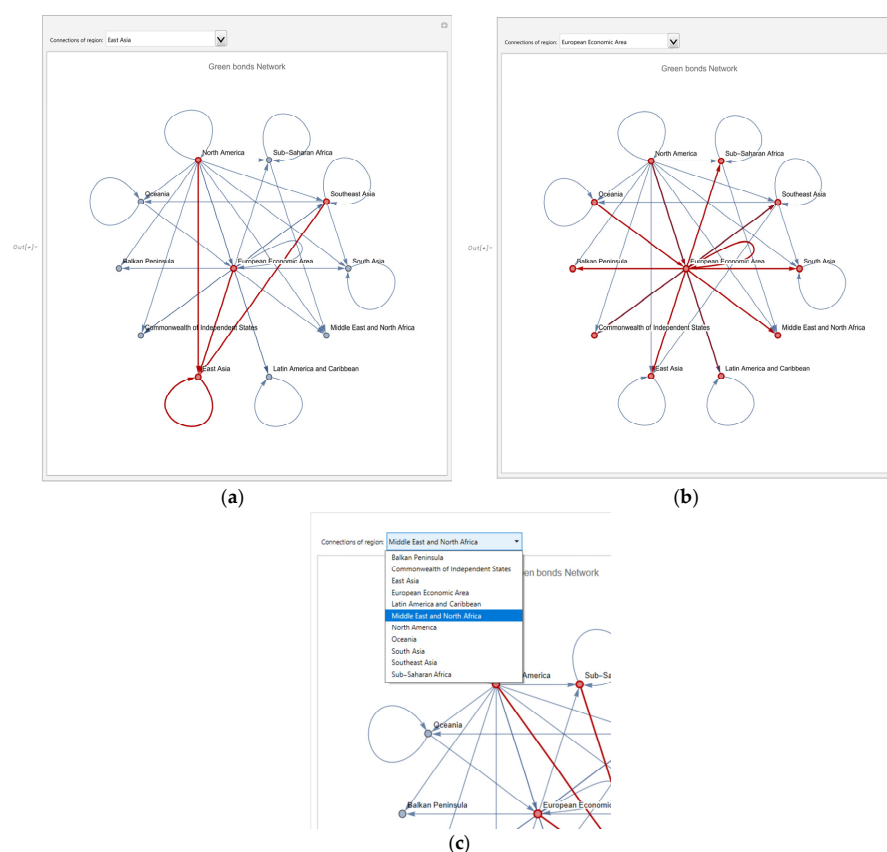


Figure 2. Connectivity of regions in the global green bond network. Connections are the transactions between regions. The connections of the selected region in the drop-down list are marked in red. (a) East Asia international transactions over 2006–2017 in the background of the global green bond transfer network; (b) European Economic Area international transactions over 2006–2017 in the background of the global green bond transfer network; (c) Manipulate control for the selected node.

The flowchart of Figure 3 summarizes the evaluation procedure of the Manipulate command, used for the interactive output of Figure 2. The sequential steps are described by Program 1 below.

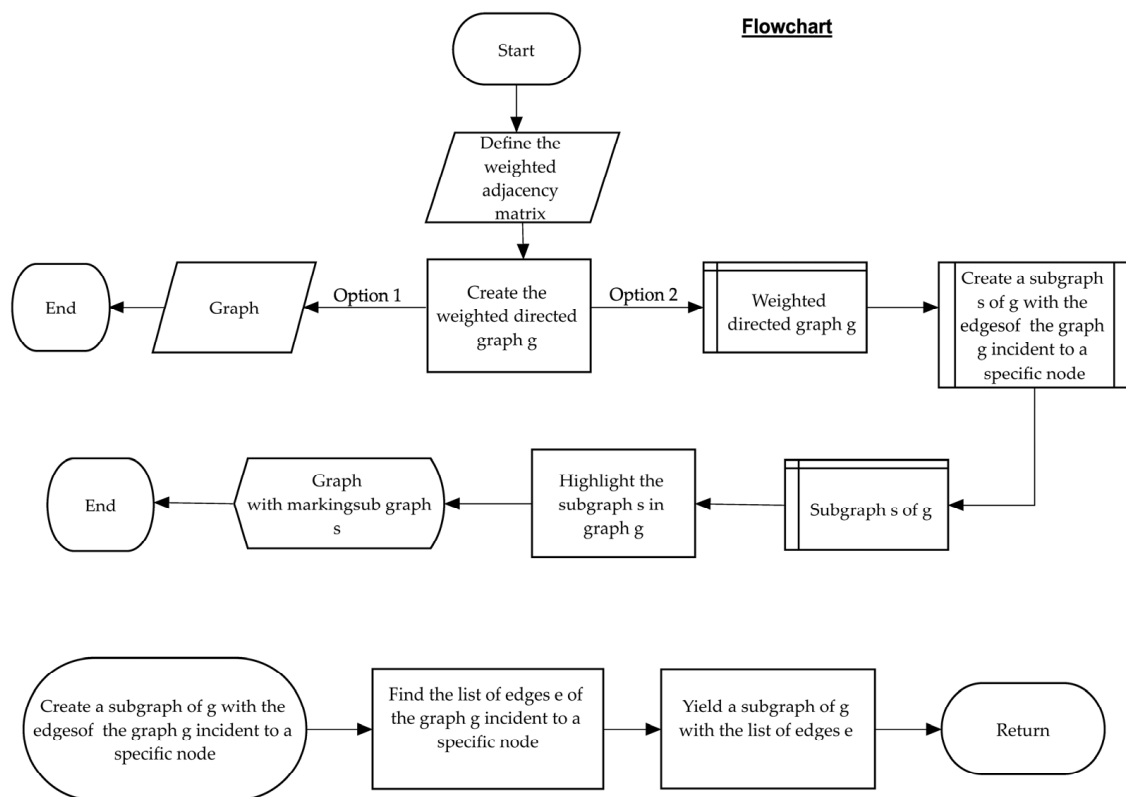


Figure 3. The program implementation in Mathematica for a graph that highlights the connections of selected nodes (as in Figure 2).

Program 1

1. Input the weighted adjacency matrix representing the graph (following the instructions in Section 2.2);
2. Create the graph g with all counterparties as nodes with the `WeightedAdjacencyGraph` function;
3. Find the list of edges of the graph g (as created in step 2) incident to a specific node, using the `IncidenceList` function;
4. Yield a subgraph of g with the edges of step 3 using the `Graph` function;
5. Highlight the subgraph of step 4 using the `HighlightGraph` function;

The evaluation part of the Manipulate command for analyzing the scenario of excluding one counterparty is summarized in the flowchart of Figure 4 and the sequential steps of Program 2 below. The output is depicted in Figure 5. The Mathematica codes are given in the Supplementary Material.

Program 2

1. Input the weighted adjacency matrix representing the network;
2. Create the graph g with all counterparties as nodes using the `WeightedAdjacencyGraph` function;
3. Create a graph by deleting a specific region and all edges connected to this region from graph g (as created in step 2) using the `VertexDelete` function;

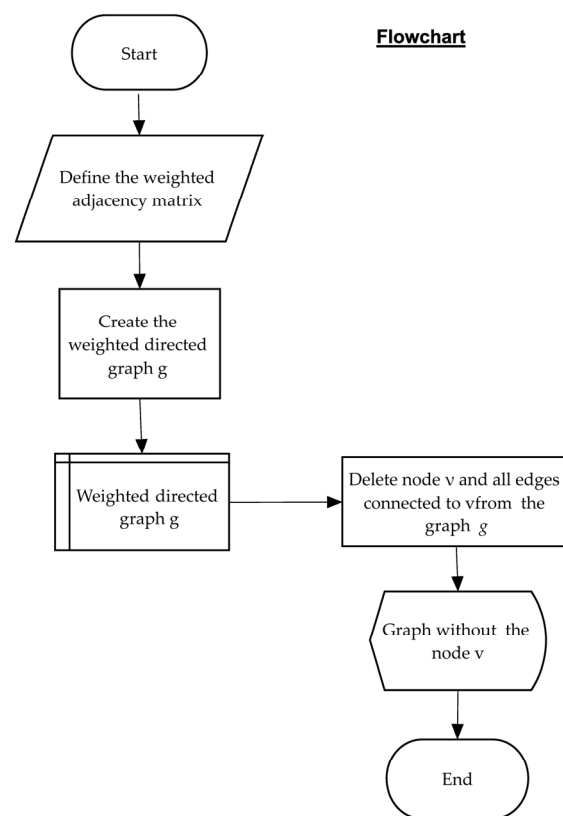


Figure 4. The program implementation in Mathematica for a graph without certain node(s) as in Figure 5.

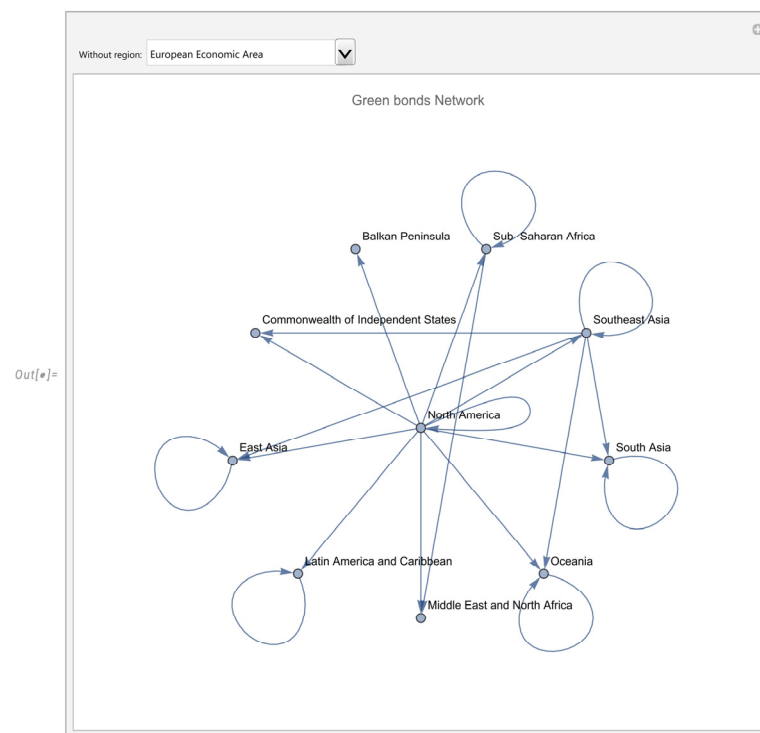


Figure 5. What if scenario analysis: the green bond network, excluding the European Economic Area. The excluded region is selected from a drop-down list at the top left corner of the output cell.

In pursuit of a full, in-depth evaluation of financial flows, we propose a dashboard of different visual results aiming to reproduce the financial network and identify the biggest providers and receivers of financial investments. Three views are provided for the analysis of the financial interactions: the heatmap of the weighted adjacency matrix and two network visualizations, with one indicating the financial allocators' perspective and one indicating the financial receivers' perspective. The dashboard is an interactive object adjusted by the selection of the year of the study period in a drop-down list.

The Manipulate command for a dashboard with (1) the heatmap of the weighted adjacency matrix and (2) a two-sided view of the structure of the network highlighting nodes according to their strength-in centrality and strength-out centrality has a programming structure with a series of nested functions. The tasks performed here are presented in the flowchart of Figure 6 and the sequential steps of Program 3 below. The Mathematica codes are given in the Supplementary Material.

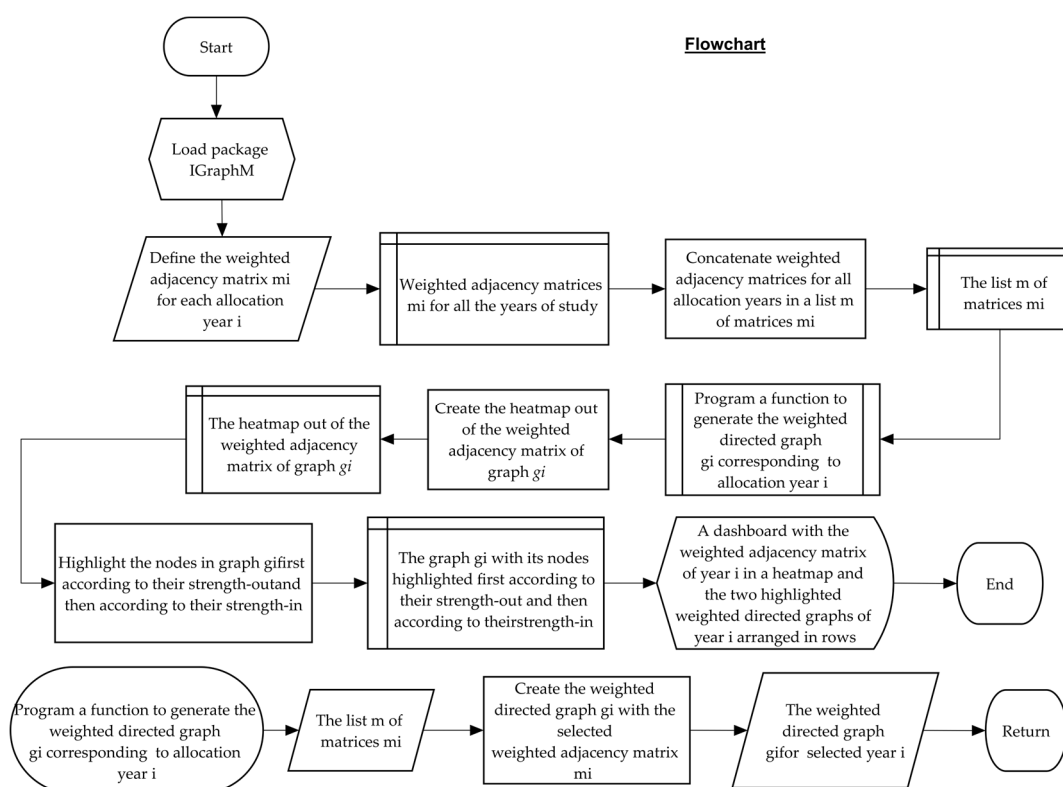


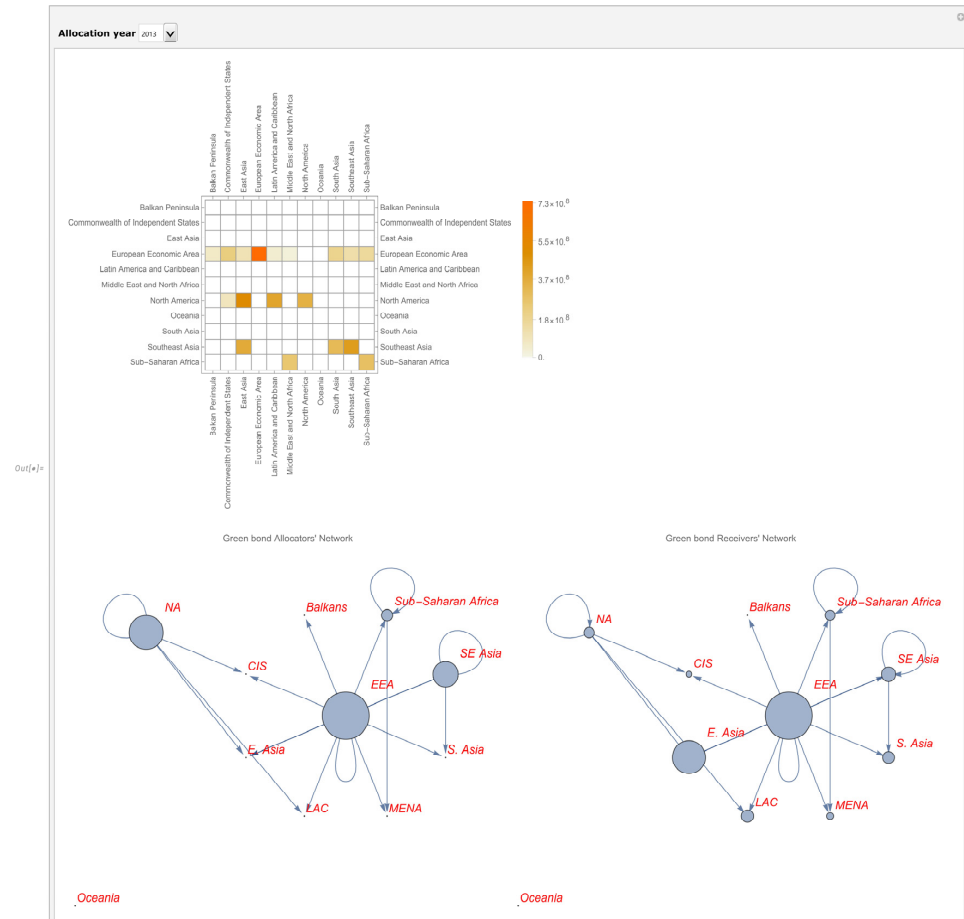
Figure 6. The program implementation in Mathematica for a dashboard with (1) the heatmap of the weighted adjacency matrix and (2) a two-sided view of the structure of the network (according to the strength-in centrality and strength-out centrality of the nodes) as in Figure 7.

Program 3

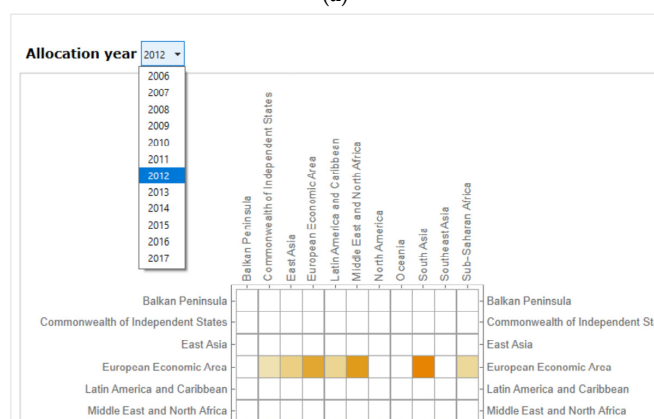
1. Load package IGraphM;
2. Input the weighted adjacency matrices for each allocation year;
3. Concatenate weighted adjacency matrices for all allocation years using the Join function;
4. Create the graph g from each weighted adjacency matrix corresponding to each allocation year using the IGWeightedAdjacencyGraph function;
5. Plot the weighted adjacency matrix of graph g (as created in step 4) using the IGAdjacencyMatrixPlot function;
6. Scale nodes of the graph g by their strength using the IGVertexMap function; IGVertexStrength returns the strength of each node, i.e., the total edge weight of edges

adjacent to it. In- and out-strength can be calculated separately for directed graphs by the IGVertexOutStrength and IGVertexInStrength functions;

7. Arrange the objects of steps 4, 5, and 6 in a row, potentially extending over several lines, using the Row function.



(a)



(b)

Figure 7. (a) Up: A heatmap illustrates the allocators (given in rows) and the receivers (given in columns) pattern for allocation proceeds. Dots in the color legend denote the decimal points. Down: The region-to-region green bond transfer network with a tooltip option for edge weights. In the green bond allocators' network (left), node size is proportional to their strength-out. In the green bond receivers' network (right), node size is proportional to their strength-in. (b) The allocation year is selected from a drop-down list at the top left corner of the output cell.

An equivalent output of Figure 7 using solely built-in Mathematica functions (i.e., without the use of the IGraph/M package) can be achieved with the program depicted in the flowchart of Figure 8. The sequential steps are given in Program 4 below. The expected output is depicted in Figure 9. The Mathematica codes are given in the Supplementary Material.

Program 4

1. Input the weighted adjacency matrices for each allocation year.
2. Concatenate weighted adjacency matrices for all allocation years using the Join function.
3. Concatenate graphs for all allocation years using the Join function.
4. Create the graph g with all counterparties as nodes and using the weighted adjacency matrix using the WeightedAdjacencyGraph function.
5. Program functions weightedout (weightedin) to calculate node strength-out (strength-in). weightedout (weightedin) takes as argument k the k -th year of the study period. Node strength s_i sums up the weights of the edges of a given node i . The strength-in of node i is expressed as in Equation (1) and the strength-out of node i as in Equation (2).

$$s_i^{in} = \sum_j^N w_{ji}, \quad (1)$$

$$s_i^{out} = \sum_j^N w_{ij}, \quad (2)$$

where w_{ij} is an entry of the weighted adjacency matrix; w_{ij} is the weight of a directed edge from node i to node j . If there is no edge, the weight is taken to be 0 [2,20].

6. Present each weighted adjacency matrix (created in step 1) using the TableForm function.
7. Highlight the nodes of graph g (as created in step 4) using the HighlightGraph function with the option VertexSize that specifies the size used for nodes (here according to their strength-out).
8. Arrange the objects of steps 6 and 7 in a column using the Column function.

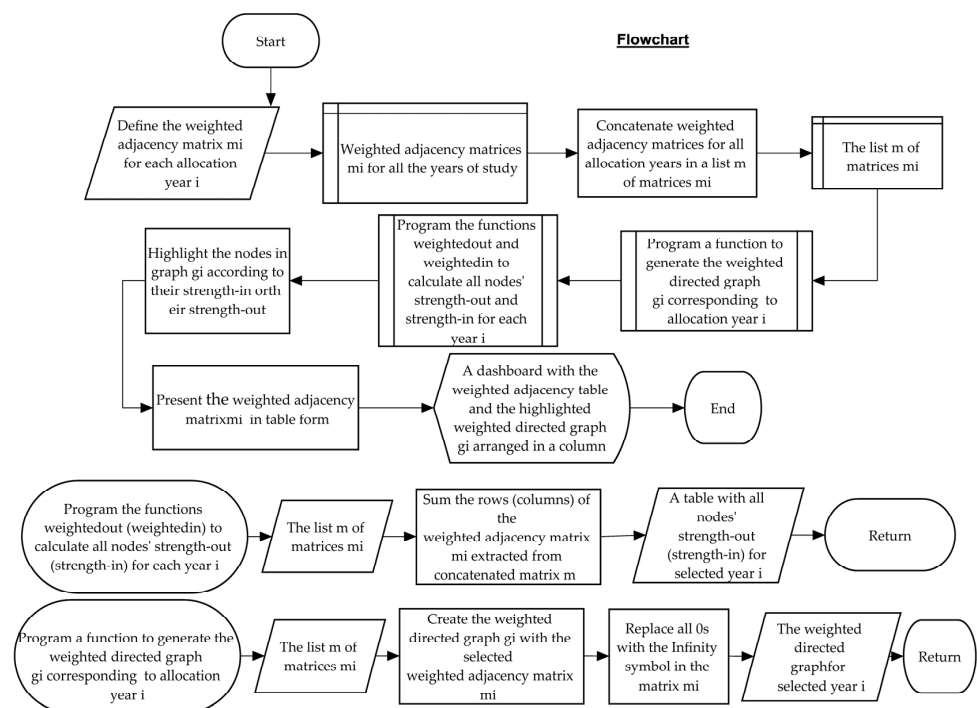


Figure 8. The program implementation in Mathematica for a dashboard with (1) the weighted adjacency matrix and (2) the network of financial flows with nodes' size proportional to their strength-out centrality (as in Figure 9).

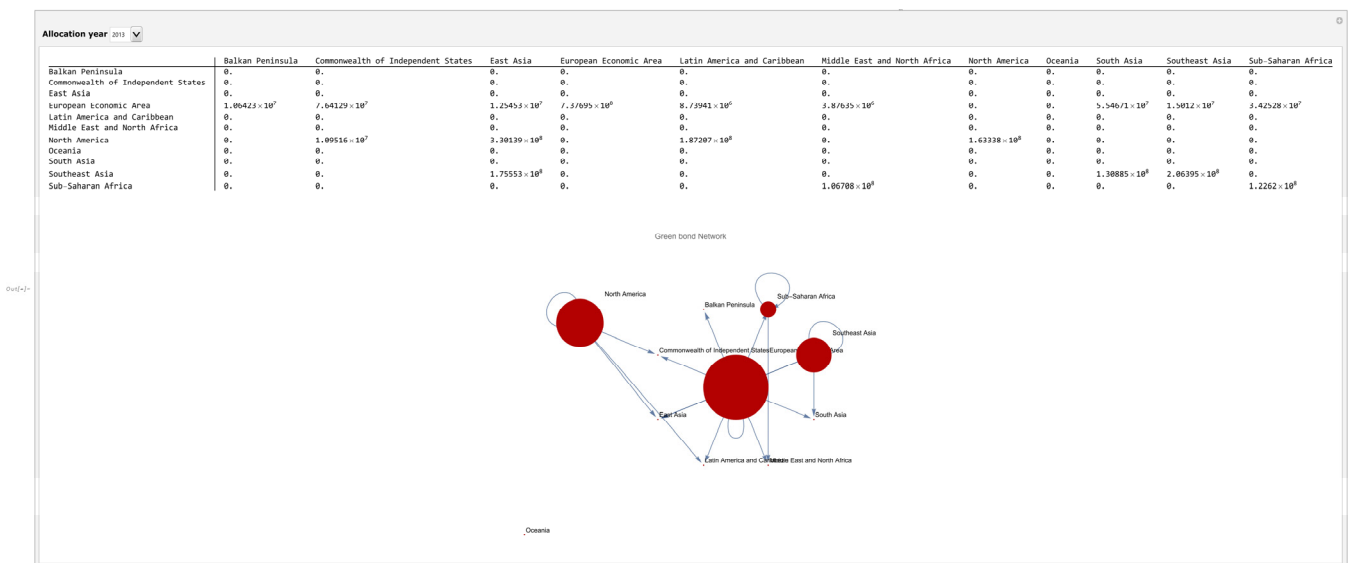


Figure 9. An interactive dashboard for the green bond market. The allocation year is selected from a drop-down list at the top left corner of the output cell. Yearly allocators' activity is reflected in the weighted adjacency matrix and the green bond allocators' network. The size of the node is analogous to its strength-out. The bigger the size, the more financial flows are transferred from the allocator region to recipient regions.

4. Building a Visualization App

For non-Mathematica users, the interactive visualization of Figure 9 (commonly referred to as “Demonstration” in Wolfram terminology) can be reproduced with the free Wolfram Computable Document Format (CDF) Player (<https://www.wolfram.com/player/>, accessed on 15 January 2024). Wolfram Player is hosted by Wolfram Research. In Wolfram Player, all demonstrations run freely on any standard Windows, Mac, or Linux computer, and there is no need to have Mathematica pre-installed.

The Mathematica codes that implement Program 4 can be inserted in a Wolfram Demonstration instead of a Mathematica Notebook. Computational steps 1–5 of Program 4 constitute the “initialization code” of the Demonstration, which is the code that must be evaluated before the Manipulate command. The Manipulate input cell (Manipulate code) incorporates steps 6–8, and the Manipulate output cell is the output in Figure 10.

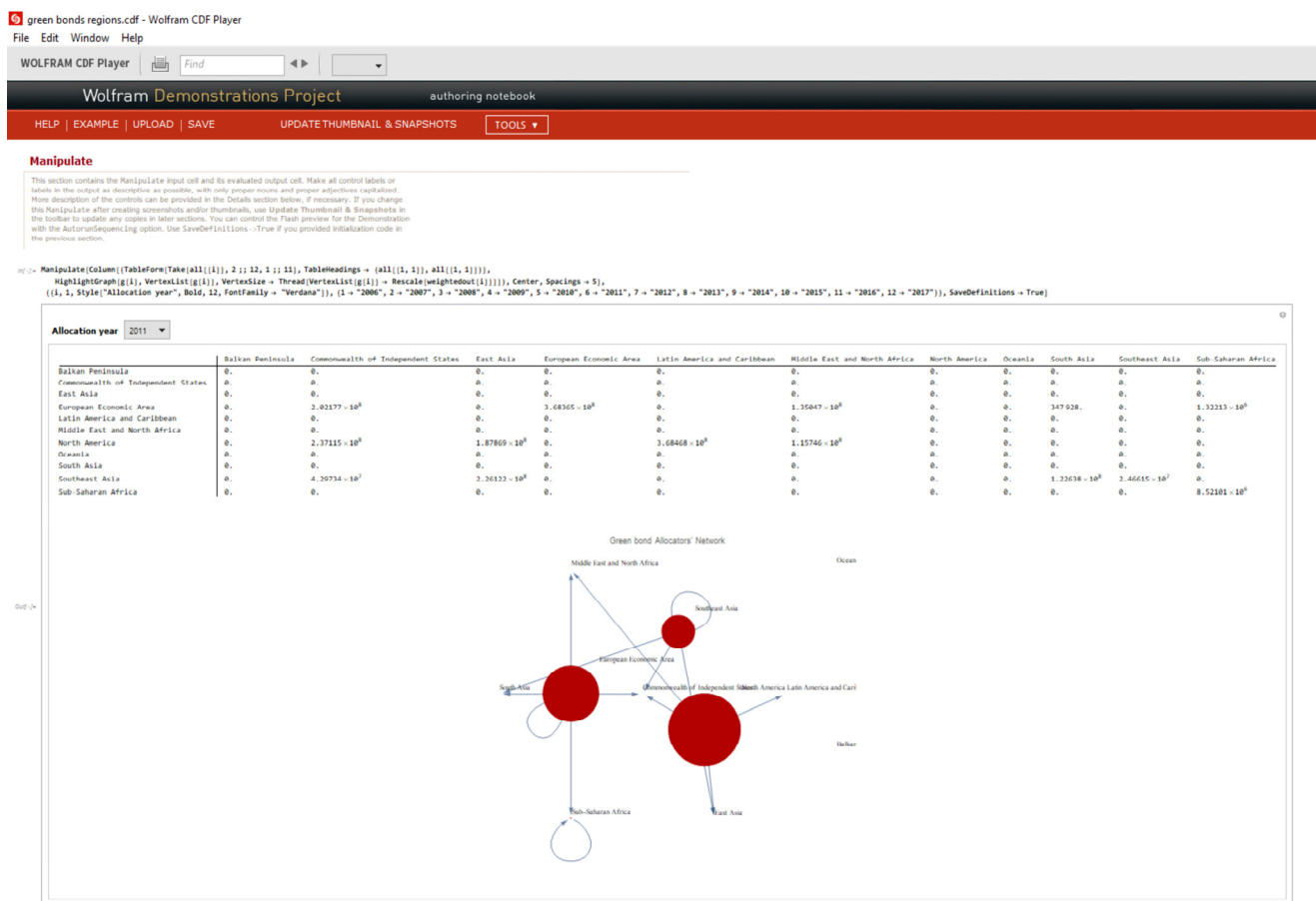


Figure 10. A screenshot from the green bond application in Wolfram Player. The application is fully functional in the free Wolfram CDF Player.

5. Results

This study sets the framework to combine mathematical thinking, reasoning, and the mathematical competencies of the Wolfram language. In this work, we achieve the following:

- (1) Develop, adapt, and customize a suite of flexible, interactive, and customizable computer models linked to existing financial data;
- (2) Develop visualizations based on a graph modeling framework to take into account the complexity of financial allocators' decision-making;
- (3) Identify the most significant members, which we consider the "backbone" of the market. The influence of counterparties is measured in terms of their financial exports' and imports' volume and frequency;
- (4) Identify market features through cognitive images for the frequency and volume of inter- and intra-regional interactions;
- (5) Develop an application that serves analysts to investigate how practices, patterns, and roadmaps of any financial market evolve, from a network perspective;
- (6) Present self-explanatory results.

Programming modules in Wolfram Language require as inputs the weighted adjacency matrices for each allocation year or the concatenation of yearly weighted adjacency matrices for all allocation years. The graph resulting from each weighted adjacency matrix corresponds to each allocation year.

The outputs include a variety of representations derived from graph models with interactive representations and highlights on certain features (such as node centralities, the list of edges incident on a selected node), and interactive dashboards that include the

weighted adjacency table in tabular form or as a heat map for each selected year, and two views of the same weighted directed and aggregated network, one with its node sizes proportional to the strength-in centrality and the other with its node sizes proportional to the strength-out centrality.

6. Discussion

There are several digital tools available that allow users to create interactive network visualizations. The most popular ones are the open-source network analysis and visualization software Gephi (v. 0.10.1), the open-source software platform for visualizing complex networks Cytoscape, JavaScript libraries Sigma.js, D3.js, Graphistry platform, JavaScript toolkit KeyLines, and the graph visualization platform Linkurious. These tools are valuable for exploring and analyzing relationships, connections, and patterns within complex datasets.

The widely known open-source software Gephi provides interactivity with the options of focusing on a certain entity, clicking a node and seeing its connections, and zooming in and out. There is also the capability of building dynamic/temporal graphs where Gephi can automatically detect if you are working with a dynamic graph and will show the “Enable Timeline” feature at the bottom of the window. This timeline shows that the system being visualized exists between a selected time interval.

Interactive network customization is also feasible in R, where dedicated packages foster some options that allow one to customize the node, links, and label feature, and other options allow one to control the layout, i.e., how nodes are distributed on the window. Some main packages for interactive network visualizations in R are visNetwork (maintained by Benoit Thieurmél), threejs (maintained by Bryan W. Lewis), NetworkD3 (maintained by Christopher Gandrud), and ndtv (maintained by Skye Bender-deMoll) [21]. The networkD3 package allows the building of interactive network diagrams with R, with the option to hover a node and drag it to see how it works and scroll to zoom in and out. The ndtv package is built to “render temporal network data as movies, interactive animations, or any other representations of changing relational structures and attributes”.

Comparing the level of interactivity, and the specific features and functionalities offered by each tool with the Mathematica software, it is obvious that Wolfram language avoids the use of a selected counterparty as a drawing component (for example, see what happens if we click on a node). Mathematica goes beyond typical interactivity controls (highlight, collapsed nodes, selection, zoom, movement of nodes, tooltip). In Mathematica, the interactive mapping system is itself a form of symbolization. Choosing which modes and functions of interaction to include can facilitate an understanding of information represented on interactive network charts and provides insights into transactions, dependencies, and interactions. “Dynamic network” (or “temporal network”) building is feasible in Mathematica with snapshots of the network over time (see Figures 7, 9 and 10).

Examples of applications of the proposed interface in real-world financial analysis are banking networks with banks as nodes and banking transactions as edges, networks in international banking with countries as nodes and capital flows as edges, and bitcoin transactions with BTC wallets as nodes and transactions as edges. Other fields of application are stock market networks, financial networks formed by the interaction of different financial entities (individual traders, banks, financial companies, financial instruments, countries, etc.) [22], networks of control such as, for example, the ownership network and the Board of Directors network, and trading networks such as the World Trade Web and the Banks’ Credit networks [23]. The proposed visualizations could adequately represent capital flows between companies as well.

7. Installation Tips

For non-Mathematica users, Wolfram Player is available freely from <https://www.wolfram.com/player/> (accessed on 15 January 2024) to view and interact with our interactive computable

documents (<https://reference.wolfram.com/language/howto/DeployANotebookForPlayer.html>, accessed on 15 January 2024).

The IGraph package for Mathematica (referred to as IGraph/M) comes pre-packaged and ready to use on Windows, OS X, and Linux (64-bit Intel), as well as the Raspberry Pi computer (<http://szhorvat.net/pelican/igraphm-a-mathematica-interface-for-igraph.html>, accessed on 15 January 2024). The code is open source and can be compiled on other platforms as well (such as 32-bit Windows). The simplest way to automatically download and install the latest release of IGraph/M is to evaluate the following line in Mathematica:

```
Get["https://raw.githubusercontent.com/szhorvat/IGraphM/master/IGInstaller.m"]
```

Once installed, the package can be loaded using <<IGraphM'.

8. Conclusions

This work explains and illustrates flows, interactions, and roles in financial markets in a more informal, graphical, and accessible way. The Mathematica-based graph models can identify financial market features and provide cognitive images for the frequency and volume of inter- and intra-regional interactions.

Interactivity in developing business models allows their users to investigate how practices, patterns, and roadmaps evolve from a network perspective. Mathematica software can help users explore and understand their data by creating interactive visualizations. The Wolfram language provides state-of-the-art functionality for modeling, analyzing, synthesizing, and visualizing graphs and networks, which is further implemented by the IGraph package for Mathematica.

By following the above-described procedure, it is possible to create the framework for the efficient development of more effective models that are easy to use and easy to modify. The same programming module can be reproduced in a Mathematica notebook or a Computable Document Format, as a Wolfram Demonstration, which enables the interaction without buying the software.

Supplementary Materials: The following supporting information can be downloaded at: <https://www.mdpi.com/article/10.3390/math12060877/s1>. The Mathematica codes. Mathematica 11.3 or later version is needed.

Funding: This research received no external funding.

Data Availability Statement: Data sharing is not applicable to this article as no new data were created or analyzed in this study.

Acknowledgments: The author would like to thank Vasileios Tzimourtos for his helpful and critical comments on the flowcharts of the paper.

Conflicts of Interest: The author declares no conflicts of interest.

References

1. Tolliver, C.; Keeley, A.R.; Managi, S. Green Bonds for the Paris Agreement and Sustainable Development Goals. *Environ. Res. Lett.* **2019**, *14*, 064009. [CrossRef]
2. Henning, M.A.; van Vuuren, J.H. *Graph and Network Theory an Applied Approach Using Mathematica®*; Springer: Cham, Switzerland, 2022.
3. Ufuktepe, Ü.; Bacak, G. Applying Mathematica and WebMathematica to Graph Coloring. *Future Gener. Comput. Syst.* **2007**, *23*, 716–720. [CrossRef]
4. Pemmaraju, S. *Computational Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*; Choice Reviews Online; Cambridge University Press: Cambridge, UK, 2004; p. 42. [CrossRef]
5. Skiena, S. *Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*; Addison-Wesley: Boston, MA, USA, 1990.
6. Freeman, J.A. *Simulating Neural Networks with Mathematica*; Addison-Wesley: Boston, MA, USA, 1994.
7. Csardi, G.; Nepusz, T. The Igraph Software Package for Complex Network Research. *InterJournal Complex Syst.* **2006**, *1695*, 1–9.
8. Wolfram, S. *An Elementary Introduction to the Wolfram Language*, 3rd ed.; Wolfram Media, Inc.: Champaign, IL, USA, 2023.
9. Bravo-Benitez, B.; Alexandrova-Kabadjova, B.; Martinez-Jaramillo, S. Centrality Measurement of the Mexican Large Value Payments System from the Perspective of Multiplex Networks. *Comput. Econ.* **2016**, *47*, 19–47. [CrossRef]

10. Nagurney, A.; Hughes, M. Financial Flow of Funds Networks. *Networks* **1992**, *22*, 145–161. [[CrossRef](#)]
11. Hughes, M.; Nagurney, A. A Network Model and Algorithm for the Analysis and Estimation of Financial Flow of Funds. *Comput. Sci. Econ. Manag.* **1992**, *5*, 23–39. [[CrossRef](#)]
12. León, C. *Authority Centrality and Hub Centrality as Metrics of Systemic Importance of Financial Market Infrastructures*; SSRN: Rochester, NY, USA, 2013. [[CrossRef](#)]
13. Leen, C.; Berndsen, R.; Renneboog, L. *Financial Stability and Interacting Networks of Financial Institutions and Market Infrastructures*; SSRN: Rochester, NY, USA, 2014. [[CrossRef](#)]
14. León, C.; Machado, C.L.; Cepeda, F.; Sarmiento, M. *Too-Connected-To-Fail Institutions and Payments System's Stability: Assessing Challenges for Financial Authorities*; SSRN: Rochester, NY, USA, 2012; Volume 644. [[CrossRef](#)]
15. Li, Y.; Liu, G.; Pin, P. Network-Based Risk Measurements for Interbank Systems. *PLoS ONE* **2018**, *13*, e0200209. [[CrossRef](#)] [[PubMed](#)]
16. Anufriev, M.; Panchenko, V.; Pinotti, P. *A Model of Network Formation for the Overnight Interbank Market*; SSRN: Rochester, NY, USA, 2016. [[CrossRef](#)]
17. Martino, R.; Ventre, V. An Analytic Network Process to Support Financial Decision-Making in the Context of Behavioural Finance. *Mathematics* **2023**, *11*, 3994. [[CrossRef](#)]
18. Balci, M.A.; Batrancea, L.M.; Akgüller, Ö.; Nichita, A. Coarse Graining on Financial Correlation Networks. *Mathematics* **2022**, *10*, 2118. [[CrossRef](#)]
19. Wang, M.; Hua, C.; Xu, H. Dynamic Linkages among Carbon, Energy and Financial Markets: Multiplex Recurrence Network Approach. *Mathematics* **2022**, *10*, 1829. [[CrossRef](#)]
20. Newman, M.E.J. Analysis of Weighted Networks. *Phys. Rev. E Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.* **2004**, *70*, 056131. [[CrossRef](#)] [[PubMed](#)]
21. Ognyanova, K. *Network Visualization with R*; POLNET 2015 Workshop: Portland, Oregon, 2015.
22. Kenett, D.Y.; Havlin, S. Network Science: A Useful Tool in Economics and Finance. *Mind Soc.* **2015**, *14*, 155–167. [[CrossRef](#)]
23. Battiston, S.; Glattfelder, J.B.; Garlaschelli, D.; Lillo, F.; Caldarelli, G. The Structure of Financial Networks. In *Network Science*; Estrada, E., Fox, M., Higham, D., Oppo, G.L., Eds.; Springer: London, UK, 2010; pp. 131–163.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.