KEYWORDS ▸ Product architecture ▸ Modularity ▸ Eigenvalue decomposition ▸ Spectral clustering

A SPECTRAL ANALYSIS SOFTWARE TO DETECT MODULES IN A DSM

Somwrita Sarkar
Design Lab, University of Sydney, Australia.
somwrita.sarkar@sydney.edu.au

Andy Dong
Faculty of Engineering and Information Technologies, University of Sydney, Australia
andy.dong@sydney.edu.au

ABSTRACT

This paper presents a new spectral clustering and partitioning based software tool for the identification of modules in design structure matrices. The MATLABÒ-based software can identify the globally optimal number of modules in the design structure matrix and identify overlapping and hierarchically overlapping modules. The software tool provides the capability for the modeler to vary the level of granularity of the analysis so as to obtain either a high-level or a granular view of modularity at the components, sub-systems, or system levels, the number of levels defined arbitrarily. It also provides the modeler with the flexibility of visualizing the membership of overlapping components to modules in terms of continuously varying membership strength. A link is provided for interested readers to download the software.

1. Introduction

The identification of modules in a design structure matrix *(DSM)* is a fundamental step in the application of DSMs. Research in methods for identifying modules in design structure matrices *(DSMs)* has been continuing. The basic idea underlying module detection algorithms is to cluster elements in the DSM such that elements that are highly coupled among each other but relatively weakly coupled to other elements appear together in the same module. Existing methods operate iteratively by adding or removing elements to modules and then calculating metrics that quantify the strength of the clusters or by partitioning matrices according to various dependency metrics. Because these methods are locally greedy, some methods utilize stochastic hill-climbing algorithms to identify a globally optimal clustering *(Borjesson and Hölttä-Otto, 2012; Helmer et al., 2008)*. Given the computational efficiency challenges associated with stochastic algorithms, other researchers have proposed qualitative, heuristic-based methods for identifying modules based on sub-functions related by flows *(Stone et al., 2000)*, which have been shown to be applicable to both small-scale and large-scale systems *(Day et al., 2009)*.

Associated with the problem of identifying modules is the problem of quantifying the degree of modularity of a DSM, that is, a metric to quantify how many modules exist, if any. Divergent results exist depending upon the level at which to identify modules *(part, sub-system, or function)*. In one study, the Singular Modularity Index *(SMI)* is presented as the preferred metric of modularity when assessing modularity from a functional perspective *(Hölttä-Otto and de Weck, 2007)*, but a more recent paper proposes that the Whitney Index *(WI)* is a more suitable metric for assessing modularity from a component perspective *(Van Eikema Hommes, 2008)*. Alternatively, Wang and Antonsson *(2004)* propose an information-theoretic measure of modularity that is independent of module type. At a component level, Sosa et al. *(2007)* recommend centrality-based measures of component modularity. At a sub-system level though, the same authors developed a different approach *(Sosa et al., 2003)*. The lack of a uniform metric for modularity can result in methods for producing and identifying modules to yield divergent results *(Chiriac et al., 2011; Hölttä and Salonen, 2003)*.

It is thus a rather open problem as to the appropriate method for finding modules and the appropriate metric that can provide integrated insights into the modularity of a given system *(system modularity)* or of a set of components *(component modularity)* at any desired level of abstraction, since these two problems go hand-in-hand *(Gershenson et al., 2003)*.

In this paper, we present a new software tool to identify modules in a DSM based on the spectra *(the set of eigenvalues)* of matrices, graphs, and networks. The method stems from a long history in graph theory to infer relations between the spectral and structural properties of a graph by studying the eigenvalues and eigenvectors of a matrix associated with the graph *(Biggs, 1994; Cvetković et al., 1995)*.

The organization of the paper proceeds in the following manner. First, we summarize the mathematical foundations of graph spectra methods and how they provide a global view of connectivity structure without losing local information. Second, we briefly review the mathematical proofs associated with our method for identifying modules in networks, whether unipartite or bipartite, thus making the approach suitable for application on single and multi-domain design structure matrices *(DSMs, DMMs, and MDMs)*. We then present some case studies demonstrating the use of the research software on synthetic and real world modular, hierarchical modular, overlapping modular DSMs. We conclude with potential extensions to the software to suit various DSM analyses.

2. Mathematical Foundations

Detecting modules in design structure matrices is a key step in various types of analyses *(Browning, 2001)*, including determining suitable interfaces between sub-systems *(Sosa et al., 2007)*, finding architectural platforms for products *(Gao et al., 2009)* and their associated project plans *(Xu and Jiao, 2009)*, and determining task groups to minimize the potential for rework *(Browning and Eppinger, 2002)*. The problem also occurs in matrix-based design decomposition to simplify design problem formulations *(Alfaris et al., 2010; Li, 2010)*. As such, it is a problem of broad interest and significance

in complex systems engineering. The problem of finding modules in design structure matrices shares similar properties with the general problem of finding modules in complex networks or graphs that represent social relations, biological systems, or computer networks. The basic problem is to identify communities of elements that share common properties by using only the information encoded in the graph representation. Despite deep interest in this problem across multiple disciplines, the problem is still not considered 'solved' *(Fortunato, 2010)* including the problem of identifying modules that are very small in scale relative to the network size *(Fortunato and Barthélemy, 2007)*.

Spectra and network structure

Spectral methods rely on the eigenvalues of matrix representations of networks, and capture global information on structure. The basic premise is that networks have distinct spectra and hence the spectra are 'fingerprints' of network topology *(Farkas et al., 2001)*. Researchers extended these results to study modularity and hierarchy using scale-free hierarchical modular networks *(de Aguiar and Bar-Yam, 2005)*.

We recently proved that the spectra of networks also fingerprint the number of modules, the number of hierarchical levels of modules, and the exact modular composition of a network and determines the limit of resolution for module detection *(Sarkar and Dong, 2011; Sarkar et al., 2013b)*, findings that have also been independently reported by other researchers *(Newman, 2013; Peixoto, 2013)*. Our insight is that the number of linearly independent eigenvectors is related to the number of modules in the network, an insight that has been partially explored by others *(Platanitis et al., 2012)*. We have exploited this finding to identify modules in a complex product architecture *(Sarkar et al., 2013a)*.

We briefly review the concepts behind the mathematical proof of our method, and refer the reader to the other publications for the details of the formal proof. We proved that the gaps between clusters of closely spaced large eigenvalues that are well separated from the bulk distribution of eigenvalues around the origin reveal the number of hierarchical levels and the number of modules at each hierarchical level. In addition, we proved that for L hierarchical levels, the expected values of the largest eigenvalues separated from the bulk of eigenvalues follow a specific pattern based on the hierarchical structure of the network. One of the strengths of our approach is that if there is no modular organization in the DSM, the spectra will show exactly 1 large eigenvalue well separated from the bulk; such a DSM has the properties of a random network, wherein modules do not occur other than by chance. We derived analytical expressions for the mean values of these largest eigenvalues, thus providing an algorithm and metric independent manner of characterizing the hierarchical modularity of networks. By combining these analytical derivations, we produced a generalized method for detecting the modular organization of networks at multiple hierarchical levels *(Sarkar and Dong, 2011; Sarkar et al., 2013b)*.

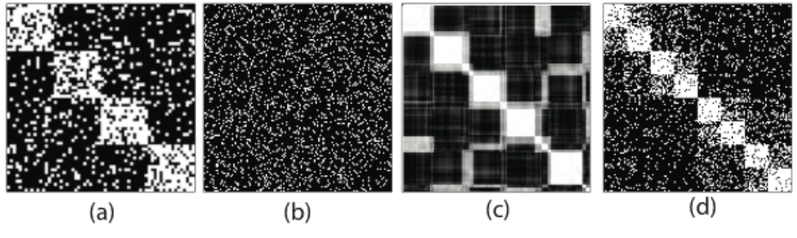


FIGURE 1: EXAMPLES OF DSMS. *(a) synthetically produced modular DSM with 64 nodes and 4 modules. (b) synthetically produced DSM with 128 nodes and 5 modules with nodes sharing overlaps with multiple modules generated using software from Lancichinetti and Fortunato (2009); original data is randomly ordered, with no modules or overlaps visible. (c) The synthetic DSM shown in (b) is input into the software we present in this paper and the output can be seen as a reordered DSM, showing the 5 modules as well as overlapping nodes between modules. (d) synthetically generated DSM with 128 nodes and 4 hierarchical levels with decreasing strength of connectivity with each level.*

Spectral analysis for modularity detection

Our spectral method for detecting modules in networks, or equivalently matrices, operates in the following manner. Here we briefly summarize the method, but for full details, see *(Sarkar and Dong, 2011; Sarkar et al., 2013a)*

First, the modeler produces a square or rectangular matrix representation of a system, that is, a single domain design structure matrix or a multi-domain design structure matrix. The values in the matrix can either be binary or positive integers *(i.e., weighted)*. **Figure 1** shows examples of a synthetically produced modular DSM, a modular DSM with overlapping modules, and a hierarchically modular DSM. In each of these examples, white matrix elements indicate a strong relationship, say 1, and elements in black represent no relation, say 0. The algorithm also accepts weighted representations, so the connectivity can be numerically graded.

Next, the software either performs an eigenvalue or singular value decomposition of the matrix depending upon whether the matrix is square or rectangular, unipartite or bipartite, respectively. Eigenvalue decomposition is used for weighted or unweighted symmetric square matrices. Singular value decomposition is used for weighted or unweighted bi-partite rectangular matrices or asymmetric square matrices. The eigenvalues are ordered in decreasing value, along with the corresponding eigenvectors.

The main idea now is to express the position of each node as a vector in space, such that the distance between two nodes in this space is a measure of the strength of their connectivity. This is done by expressing each node or component as a vector in space defined as a linear combination of the k-largest eigenvectors multiplied their corresponding k-largest eigenvalues. Thus, the software performs a dimensionality reduction on the original DSM by preserving the k largest eigenvectors and eigenvalues to produce a reduced approximation of the DSM.

The choice of the k value is guided by the findings on the separation of large eigenvalues; each k value corresponds to the number of modules at a specific level of hierarchy. In a perfectly modular network, the value of k is exactly the number of modules in the network. In practice, the choice of the value of k can be left to the choice of the modeler, wherein a small value of k results in a high-level view of the network *(i.e., k large modules)* whereas a large value results in a granular view of k smaller-sized modules. The value of k can be increased up to the limit of module detection. The choice of k is similar to the post-processing step described by Helmer et al. *(2008)* wherein modelers can merge modules. Decreasing the value of k has a similar effect.

In the final step, to identify the modules, dot products are computed between all the k reduced vector representations of nodes in the lower k-dimensional space. The higher the cosine between two node vectors, the higher the probability that they belong to the same module. We developed a simple matrix re-ordering algorithm to reveal the highly connected groups of nodes along the block diagonal. Thus, as shown in **Figure 1***(c)*, the modules reveal themselves as the white blocks off the main diagonal. Horizontally or vertically from the modules in off-diagonal positions, the elements represent inter-module connections, that is connections between nodes across modules. The more inter-module connections, the more integrated the DSM.

We have coded up the algorithm into a research tool, which we present next followed by a set of case studies on its operation with a synthetic and real-world DSM.

3. Software Operation

The software consists of a set of MATLABÒ-based functions, and is available for download by contacting the authors. It will shortly be made available via a University weblink. Four functions, and associated m-files, are provided: *(1)* cosineMeasure.m *(supporting*

file to main files; calculates the cosine value between the lower dimensional vector representations of nodes), (2) reorder.m *(supporting file to main files; re-orders matrix elements based on cosine similarity values to detect modules), (c)* communityDetect_square.m *(main file for square, symmetric DSMs),* and *(4)* communityDetect_rectangular.m *(main file for square asymmetric or rectangular DSMs).*

The following is the list of steps for operating the software:

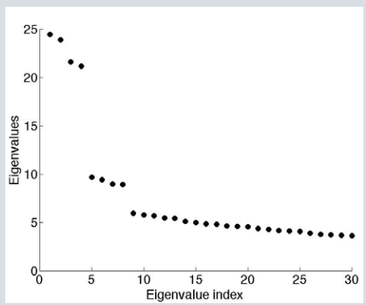
1. Input the DSM, square or rectangular in form (for single or multi-domain cases), as a MAT file.
2. Load the MAT file into MATLABÒ.
3. Run either of the two main files by first setting k = 2. For full command line syntax, use the MATLABÒ help tool.
4. The program will plot the scree plot for the eigenvalues (or singular values).
5. Observe the gaps in the spectra and choose appropriate k values with the largest gaps.
6. Re-run the main file at these k values.
7. The program outputs the modules, both in the form of the original ordering of nodes, and nodes re-ordered to show the detected modules. These matrices capture the cosine similarity values, higher values are shown in red (close to cosine values of 1), lower values are shown in blue (close to cosine values of -1). Thus, red clusters show the presence of modules.
8. Hierarchical clustering and overlapping nodes can be detected using multiple k values, to be demonstrated in the next section.

4. Case Studies

Here we demonstrate the use of the software to detect hierarchical modularity in a synthetically generated network, and overlapping modularity in the Ford Climate Control System DSM. Since we have already shown the application of the method elsewhere on much larger systems *(Dong and Sarkar, 2012; Sarkar and Dong, 2011; Sarkar et al., 2013a; Sarkar et al., 2013b),* we specifically choose the Ford Climate Control System DSM for its historical significance and small size through which we are able to demonstrate the method visually.

Hierarchical Modular DSM

In this first example, we show how the software can assist the modeler to examine the modularity of a DSM at various levels of granularity without needing to know a priori how many modules exist, if any. A synthetic hierarchical modular DSM was constructed using the stochastic block network generation model *(Sarkar et al., 2013a; Sarkar et al., 2013b).* **Figure 4***(a)* shows the DSM image. This DSM has 128 nodes and exactly three levels of hierarchy in its modular organization, with 8 modules of 16 nodes at the lowest level of granularity, 4 modules of 32 nodes at the second level, and 2 modules of 64 nodes at the third hierarchical level.



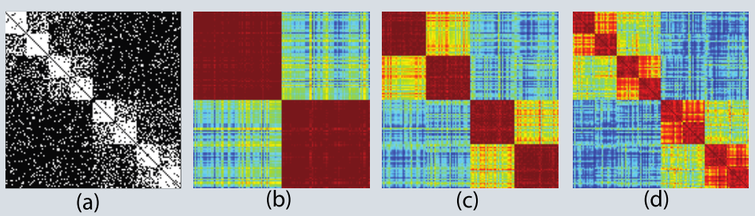


FIGURE 4. *Results of modularity detection on hierarchical DSM using software. (a) original DSM plotted using function imshow(). (b),(c),(d) Modules output from software by setting k = 2, 4, and 8, respectively.*

FIGURE 3. *Eigenvalues of the hierarchical DSM, gaps between the eigenvalues show the hierarchical structure of the DSM. (Last few eigenvalues, up to the 128, have not been shown for sake of clarity in representation).*

Figure 3 shows the plot of the eigenvalues of the DSM output by MATLABÒ. Note the large gaps between the 2nd, 4th, and 8th eigenvalues and their separation from the bulk of the spectrum. This signifies that there are 8, 4, and 2 modules at three hierarchical levels, as expected.

Now, running the code with k set to 2, 4, and 8, respectively, brings out the modules at these hierarchical levels. When the value of k is set at 2, only the high-level module is identified and visualized; as the value of k is increased, the other levels of hierarchical modularity become evident. **Figure 4** shows the original DSM and modules correctly identified at the three hierarchical levels.

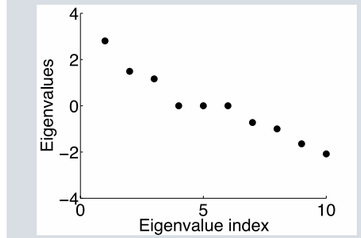
Note that choosing any other *(incorrect)* value of k around the "right" values will still result in the algorithm showing the correct number of modules. In other words, the software will not identify more modules than are actually present in the DSM, up to the limit of modularity detection. Based upon the capabilities of the software, we recommend that modelers create a DSM at the lowest level of granularity possible, such as down to the smallest practicable part or component. The software will identify the various levels of hierarchical modularity within which the part is embedded, such as a set of bolts connected to a plate *(one module)* connected to a support beam *(which, along with the plate/bolt module, can be another module).*

Real-World DSM

Figure 6*(a)* shows the Ford Climate Control System DSM *(Example 3.1, (Eppinger and Browning, 2012))*. This is a classic example, since it is often cited as the first product based DSM ever produced. The interesting feature of the Ford DSM is that it has a number of overlapping modules, and is small enough for a detailed demonstration. We use it to demonstrate the operation of the software.

The Ford DSM shows 4 types of interactions in a single DSM, showing spatial adjacency, energy flow, materials, and information flow between 16 components of a car climate control system. **Figure 5** shows the eigenvalues for the materials DSM, with only 10 of the 16 components. Note that despite the small size of the system, 3 large eigenvalues appear, signaling the appearance of 3 modules in the system. This is a consistent feature of the approach that scales to very large system sizes; we have applied the algorithm to a Pratt and Whitney aircraft engine example in *(Sarkar et al., 2013a),* and to process DSMs that go up to about 900 nodes in *(Dong and Sarkar, 2012).* We have also applied the same approach to complex networks, with number of nodes ranging up to thousands *(Sarkar and Dong, 2011; Sarkar et al., 2013b).* In each case, we have found that the number of outlying eigen or singular values, separated from the bulk of the spectrum, provides a good estimate of the actual number of modules in the system.

Figure 6 shows the results of the modularity analysis. Three main modules are identified, which are also identified in the original source literature: *(1)* Front-end air module: Components A, B, and E, *(2)* Refrigerant module: Components E, F, I, and H, and *(3)* Interior Air module: Components H, C, P, O, and G. Note that components E and H clearly

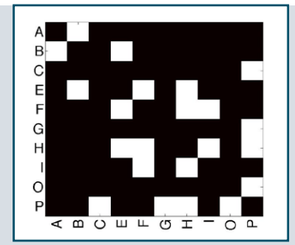


A. *Eigenvalues of the Ford Climate Control System DSM*

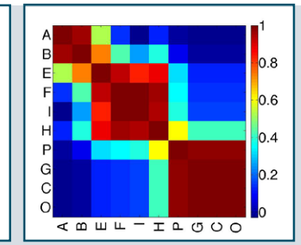
Id	Component
A	Radiator
B	Engine Fan
C	Heater Core
E	Condenser
F	Compressor
G	Evaporator Case
H	Evaporator Core
I	Accumulator
O	Blower Controller
P	Blower Motor

B. *Definition of components in the Materials DSM*

FIGURE 5.



A. *Original DSM*



B. *3 modules detected after re-ordering at nodes by setting k = 3. Red shows high connectivity, Blue shows low connectivity, overlapping nodes E and H show mid-strength connectvity to two modules.*

FIGURE 6. *Results of modularity analysis on Ford Climate Control System DSM using software.*

emerge as overlapping components shared by two modules. This is measured, as described in the algorithm and software section, as the cosines between vectors representing nodes in lower dimensional k-space, and is visually depicted using colors in **Figure 6***(b)*: red shows maximum connectivity between nodes in modules, blue shows minimum connectivity, and other shades show mid-strength connectivity *(according to color bar)*.

As is seen, component E *(Condensor)* emerges as an overlapping component between the first two modules *(Front-end air module and Refrigerant module)*, and component H *(Evaporator core)* emerges as an overlapping component between the last two modules *(Refrigerant module and Interior Air module)*.

In addition, in our analysis, components B *(Engine Fan)* and P *(Blower Motor)* also emerge as mildly overlapping with modules 2 and 3, respectively, but not to the same degree as E and H: the numerical strength of connectivity, measured in terms of the cosines, is lower but not negligible. This feature is an important part of our algorithm: it allows nodes to have continuously varying "degrees" of membership to modules, and therefore, the modeler can see that certain components can be part of modules to varying degrees and with varying strength. As a result, when the modeler makes a decision as to which component should belong to which module, that decision can be better informed by seeing the varying strengths with which components in the overlap region belong to each module. In this specific case, for example, the modeler explicitly chose component H to sit in both modules 2 and 3, but chose component P to sit only in the module 3.

5. Conclusions

We have presented a method and research software tool for the detection of modules in DSMs. In addition to being very computationally efficient, as many DSM modularity identification problems would require less than a few seconds of compute time, the tool correctly identifies the right number of modules in the DSM.

We believe that several potential enhancements to the software tool are possible. First, the matrix re-ordering algorithm is based upon a simple cosine measure of similarity. Researchers can modify this similarity calculation without affecting the operation of the software, and, in so doing, take into account particular meanings about numerical values in the DSM. We note that the numerical values in the DSM are taken to be nominal rather than categorical. Second, while we have provided a matrix based visualization mechanism to identify interface nodes by visualizing the cosine values, it would clearly be advantageous if the software could be easily extended to list the interface nodes. Finally, the software tool could be extended for use on multi-level high-definition design structure matrices *(Tilstra et al., 2012)* by incorporating the higher-order singular value decomposition *(de Lathauwer et al., 2000)* in place of the EVD/SVD on a two-dimensional DSM.

authors

Andy Dong's research addresses the central activity of engineering: the design of new products and services. He joined the University of Sydney in 2003 after completing his PhD and postdoctoral training in mechanical engineering at the University of California, Berkeley. He started in the Faculty of Architecture, Design and Planning as a Lecturer and eventually became the Head of Discipline for the Design Lab. In 2010, he was awarded an Australian Research Council Future Fellowship. He was appointed the Warren Centre Chair for Engineering Innovation in 2012 and joined the Faculty of Engineering & Information Technologies. He is on the Editorial Board of the key journals in design research including Design Studies, Journal of Engineering Design, and Research in Engineering Design.

Dr. Somwrita Sarkar's research investigates complex systems and networks in various design, technological, and biological domains, and develops computational and analytical models and methods to better understand them. She is currently working on spectral methods for graph classification, especially for modularity and hierarchy detection in complex networks. Dr. Sarkar received a Bachelor of Planning from the School of Planning and Architecture, New Delhi in 1999, a Master of Technology in Construction Engineering and Management from the Indian Institute of Technology, New Delhi in 2003, and a PhD in Design Theory and Methodology from the University of Sydney in 2009.

references

Alfaris, A., Svetinovic, D., Siddiqi, A., Rizk, C., de Weck, O. *(2010).* Hierarchical Decomposition and Multidomain Formulation for the Design of Complex Sustainable Systems. Journal of Mechanical Design 132, 091003-091003.

Biggs, N. *(1994).* Algebraic Graph Theory, 2nd ed. Cambridge University Press, Cambridge.

Borjesson, F., Hölttä-Otto, K. *(2012).* Improved Clustering Algorithm for Design Structure Matrix, ASME 2012 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. ASME, New York, pp. 921-930.

Browning, T.R. *(2001).* Applying the design structure matrix to system decomposition and integration problems: a review and new directions. Engineering Management, IEEE Transactions on 48, 292-306.

Browning, T.R., Eppinger, S.D. *(2002).* Modeling impacts of process architecture on cost and schedule risk in product development. Engineering Management, IEEE Transactions on 49, 428-442.

Chiriac, N., Holtta-Otto, K., Lysy, D., Suh, E.S. *(2011).* Level of Modularity and Different Levels of System Granularity. Journal of Mechanical Design 133, 101007-101010.

Cvetković D.M., Doob, M., Sachs, H. *(1995).* Spectra of graphs : theory and applications. Johann Ambrosius Barth Verlag, Heidelberg.

Day, R., Stone, R.B., Lough, K.G. *(2009).* Validating Module Heuristics on Large Scale Products, ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference *(IDETC/CIE2009)*. ASME, San Diego, pp. 1079-1087.

de Aguiar, M.A.M., Bar-Yam, Y. *(2005).* Spectral analysis and the dynamic response of complex networks. Physical Review E 71, 016106.

de Lathauwer, L., de Moor, B., Vandewalle, J. *(2000).* A multilinear singular value decomposition. SIAM Journal on Matrix Analysis and Applications 21, 1253-1278.

Dong A. and Sarkar, S. *(2012).* Generalized design knowledge and higher order singular value decomposition, in: Gero, J.S. *(Ed.)*, Design Computing and Cognition, 2012. Springer.

Eppinger, S.D., Browning, T.R. *(2012).* Design Structure Matrix Methods and Applications. MIT Press, MIT.

Farkas, I.J., Derényi, I., Barabási, A.-L., Vicsek, T. *(2001).* Spectra of "real-world" graphs: Beyond the semicircle law. Physical Review E 64, 026704.

Fortunato, S. *(2010).* Community detection in graphs. Physics Reports 486, 75-174.

Fortunato, S., Barthélemy, M. *(2007).* Resolution limit in community detection. Proceedings of the National Academy of Sciences 104, 36-41.

Gao, F., Xiao, G., Simpson, T. *(2009).* Module-scale-based product platform planning. Research in Engineering Design 20, 129-141.

Gershenson, J.K., Prasad, G.J., Zhang, Y. *(2003).* Product modularity: Definitions and benefits. Journal of Engineering Design 14, 295 - 313.

Helmer, R., Yassine, A., Meier, C. *(2008).* Systematic module and interface definition using component design structure matrix. Journal of Engineering Design 21, 647-675.

Hölttä, K.M.M., Salonen, M.P. *(2003).* Comparing Three Different Modularity Methods, 15th International Conference on Design Theory and Methodology. ASME, Chicago, pp. 533-541.

Hölttä-Otto, K., de Weck, O. *(2007).* Degree of Modularity in Engineering Systems and Products with Technical and Business Constraints. Concurrent Engineering 15, 113-126.

Lancichinetti, A., Fortunato, S. *(2009).* Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. Physical Review E 80, 016118.

Li, S. *(2010).* Methodical Extensions for Decomposition of Matrix-Based Design Problems. Journal of Mechanical Design 132, 061003-061003.

Newman, M.E.J. *(2013).* Spectral methods for community detection and graph partitioning. Physical Review E 88, 042822.

Peixoto, T.P. *(2013).* Eigenvalue Spectra of Modular Networks. Physical Review Letters 111, 098701.

Platanitis, G., Pop-Iliev, R., Barari, A. *(2012).* Development of a DSM-Based Methodology in an Academic Setting. Journal of Mechanical Design 134, 021007-021007.

Sarkar S. and Dong A. *(2011).* Community detection in graphs using singular value decomposition. Physical Review E 83, 046114.

Sarkar S., Dong A., Henderson, J.A., and Robisnon, P.A. *(2013a).* Spectral characterization of hierarchical modularity in product architectures. Journal of Mechanical Design 136, 011006.

Sarkar, S., Henderson, J.A., and Robinson, P.A. *(2013b).* Spectral characterization of hierarchical modularity and limits of modularity detection. PLoS ONE 8, e54383.

Sosa, M.E., Eppinger, S.D., Rowles, C.M. *(2003).* Identifying Modular and Integrative Systems and Their Impact on Design Team Interactions. Journal of Mechanical Design 125, 240-252.

Sosa, M.E., Eppinger, S.D., Rowles, C.M. *(2007).* A Network Approach to Define Modularity of Components in Complex Products. Journal of Mechanical Design 129, 1118-1129.

Stone, R.B., Wood, K.L., Crawford, R.H. *(2000).* A heuristic method for identifying modules for product architectures. Design Studies 21, 5-31.

Tilstra, A.H., Seepersad, C.C., Wood, K.L. *(2012).* A high-definition design structure matrix *(HDDSM)* for the quantitative assessment of product architecture. Journal of Engineering Design 23, 764-786.

Van Eikema Hommes, Q.D. *(2008).* Comparison and Application of Metrics That Define the Components Modularity in Complex Products, ASME 2008 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference *(IDETC/CIE2008)*. ASME, Brooklyn, pp. 287-296.

Wang, B., Antonsson, E.K. *(2004).* Information Measure for Modularity in Engineering Design, ASME 2004 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference *(IDETC/CIE2004)* ASME, Salt Lake City, pp. 449-458.

Xu, Q., Jiao, J.R. *(2009).* Design Project Modularization for Product Families. Journal of Mechanical Design 131, 071007-071007.