Computing Core/Periphery Structures and Permutation Tests for Social Relations Data*

John P. Boyd†, William J. Fitzgerald, and Robert J. Beck

University of California at Irvine, CA 92697, USA

Abstract

The core/periphery structure is ubiquitous in network studies. The discrete version of the concept is that individuals in a group belong to either the core, which has a high density of ties, or to the periphery, which has a low density of ties. The density of ties between the core and the periphery may be either high or low. If the core/periphery structure is given *a priori*, then there is no problem in finding a suitable statistical test. Often, however, the structure is not given, which presents us with two problems, searching for the optimal core/periphery structure, and devising a valid statistical test to replace the one invalidated by the search. *UCINET* (Borgatti, Everett, and Freeman, 2002), the oldest and most trusted network program, gives incorrect answers in some simple cases for the first problem and does not address the second. This paper solves both problems with an adaptation of the Kernighan-Lin search algorithm, and with a permutation test incorporating this algorithm.

Keywords: Core; Periphery; Permutation test; Algorithm

* This material is based on work supported under the United States Department of Education Grant # PT3 447625-23470-03.

† Corresponding author. Tel. +1-949-824-5427; fax +1-949-824-8651
E-mail address: jpboyd@uci.edu (J. P. Boyd)

1. Introduction

The idea that some groups or organizations have core/periphery structures has enjoyed considerable historical mention in social network analysis (e.g., Laumann and Pappi, 1976; Alba and Moore, 1978; Mintz and Schwartz, 1981). This notion continues to be employed in studies including scientific citation networks (Doreian, 1985), world economy (Smith and White, 1992), corporate structure (Barsky, 1999), and small groups (Beck et al., 2003; Beck et al.; ND; Cummings and Cross, 2003).

Borgatti and Everett (1999) presented several formal models for core/periphery structures, which were incorporated into the most widely used network analysis program, *UCINET* (Borgatti, Everett, and Freeman, 2002), for general application. Given a square data matrix, *UCINET* finds a core/periphery structure in two possible ways: either it computes the degree of "coreness" for each node or actor in a matrix (continuous model), or it finds a labeled bipartition of these nodes or actors, core and periphery subgroups (discrete model). We shall restrict our attention in this paper to the discrete model, revisiting the continuous model in another paper.

The output for each model in *UCINET* also includes an overall measure of "fitness" that indicates how well the observed data approximates an ideal core/periphery structure. A high fitness measure implies a good agreement with the model, while a lower fitness measure suggests that the model should be rejected. However, since there is no test for the statistical significance of fitness, *UCINET* users are left without the benefit or comfort of a *p*-value.

In other words, as Borgatti and Everett (1999) point out, this overall fitness value does not tell us the likelihood of obtaining, by chance alone, a value as high as the one actually observed. This is a problem when the core/periphery bipartition is not given *a priori*, but is instead selected to be optimal, or even close to optimal. This is similar to the statistical problem of determining the difference between two groups that is the output from a clustering program—obviously, running a simple *t*-test would be wrong. Although a solid statistical underpinning for these optimal core/periphery models would significantly strengthen both their usefulness and substantive interpretation, Borgatti and Everett (1999) did not specify a statistical test of significance. They maintain that this is because it is necessary to first have a theory about how ties are formed, and then to generate a null model specific to the substantive context and to the type of data at hand. Therefore, because each unique dataset demands its own null model, a universal permutation test would not be appropriate (1999:393–394).

From a pragmatic standpoint, this situation leaves researchers somewhat frustrated. The concept of a core/periphery structure is one of a number of viable *a priori* models which, if consistent with data, may have important substantive consequences and theoretical implications. Thus, we would like a more decisive and efficient way to assess the basic applicability of this model than is currently available. At this initial stage, it would be desirable to have some kind of generic permutation test that could be employed in most, if not all, situations to obtain the significance of an observed fitness value. With this solution, we would readily be able to tell whether we should move on to other models, or if it might be more fruitful to develop more specific and more powerful statistical tests based on models of tie formation as called for by Borgatti and Everett (1999). In other words, it is better to have a statistical test for even a simple null

hypothesis, even though it may not be as complex as one would like. We will point out possible directions for more complex models in the discussion section.

In this paper, we propose and develop such a universal permutation test for the discrete core/periphery model. We believe that this test would be useful in most situations where researchers need a ready assessment of whether their data are consistent with a discrete core/periphery structure. The general idea is to compare the fitness f_0 of the original data with a large number of fitness values, f_1,K ,f_N, obtained from permutations of the off-diagonal elements of the data matrix. If f_0 is larger than 95 percent of the fitness values of the permuted matrices, then the null hypothesis would be rejected.

To develop this permutation test, however, we first need to know that the genetic search algorithm (Goldberg, 1989) implemented in *UCINET* actually finds the optimum core/periphery bipartition. In the help files for this routine, Borgatti, Everett and Freeman (2002) stress that this algorithm may find only local maxima, and suggest multiple runs with different starting configurations. Failure to find the optimum solution for f_0 can increase the likelihood of a Type II error. Thus, we may decide that our data do not exhibit a core/periphery structure when in fact they do. The best way to minimize the likelihood of a Type II error is to be sure that the search algorithm actually finds the global maximum. Even if the global optimum cannot be guaranteed, then we should at least ensure that the search algorithm is close to optimal and that it performs as well on the permuted data as on the original data.

In the next section, we will briefly review the alternative fitness criteria for the discrete core/periphery model. Next, with the help of an exhaustive search algorithm, we will compare fitness values from smaller data sets using *UCINET* and three other algorithms. We include an alternative implementation of the genetic algorithm (Goldberg, 1989), a simulated annealing algorithm (Kirkpatrick, et al., 1983), and the Kernighan-Lin algorithm (Kernighan and Lin, 1970; Aarts and Lenstra, 1997; see our Appendix). We will make similar comparisons, excluding exhaustive search, using larger datasets. Finally, we will present and discuss our proposed permutation test.

2. *UCINET* discrete core/periphery fitness measures

In *UCINET*, the genetic algorithm for finding discrete core/periphery bipartitions has five fit function options. Depending upon your choice of fit function, the algorithm will try to minimize or maximize the value of the associated fitness measure.[1]

The default categorical fit function is the "CORR" option, which we will focus on for the remainder of this paper. In this case, the genetic algorithm attempts to maximize the Pearson product moment correlation between the permuted observed data matrix and an appropriate ideal core/periphery pattern matrix. The pattern matrix has ones in the core block and zeros in the periphery block. If the density of core/periphery and periphery/core blocks is specified, then this value will be placed in all cells of these off-diagonal blocks in the ideal matrix.[2]

Alternatively, treating these off-diagonal blocks as missing, as is recommended by Borgatti and Everett (1999), results in missing values in these cells. We agree that this

[1] Four of the five fit functions are briefly explained in the help files. The "HAMMING" option is not mentioned.

[2] It is not clear that this specification is meaningful for valued data.

is theoretically and substantively the most sensible approach and we use it in the following sections. In this context, the correlation between the observed permuted data matrix and the ideal matrix is focused exclusively upon the defining characteristics of what we mean by a core/periphery structure, allowing for the clearest interpretation of the fit function.[3] That is, we attempt to find the core/periphery bipartition that simultaneously maximizes connectivity in the core block and minimizes connectivity in the periphery block, while ignoring off-diagonal blocks.

These two assumptions, using the correlation function and ignoring the off-diagonal blocks, imply that if the correlation fitness for a given core/periphery structure is f, then the fitness for the structure that switches all core/periphery labels is $-f$.

3. Comparing algorithms where the global optimum is known

A conclusive way to assess whether a search algorithm actually finds the global optimum is to perform an exhaustive search and compare these results to those from the search algorithm. This method becomes impractical for large matrices because the number of possible labeled bipartitions (core/periphery structures) increases exponentially with the linear dimension of the matrix. More precisely, the number of nontrivial (where both the core and the periphery have at least two members) labeled bipartitions for an n by n matrix is $2^n - 2n - 2$. The exponential term, 2^n, corresponds to the number of subsets (not bipartitions, since one must specify which block is the core), while the negative terms exclude cores or peripheries with fewer than two points. However, this problem is tractable with smaller datasets. For example, if $n = 8$, the number of possible nontrivial labeled bipartitions is 238. If we perform exhaustive searches on small datasets to obtain the best core/periphery bipartition, as measured by maximizing the Pearson product moment correlation, we can easily compare these results to the core/periphery bipartitions found by the genetic algorithm in *UCINET* which uses the same fit function.

We will illustrate this comparison using 12 pre-service teacher online discussion groups, each with eight participants. This dataset consists of 12 eight by eight nonnegative integer valued matrices, which are not symmetric, and where the diagonal elements are undefined. These matrices represent all the messages sent and received between group members over a set period of time. The experimental procedures and substantive interpretations for these data can be found in Beck et al. (2003) and Beck et al. (ND).

The best way to compare heuristic algorithms is to establish the correct answers via an algorithm, no matter how inefficient, known to be correct. The simplest such algorithm is an exhaustive search of all the possible core/periphery structures to establish the global optimum core/periphery bipartition for each of these matrices. To accomplish this task, we programmed an exhaustive search in *Mathematica* (Wolfram, 2003). The highest matrix correlation and the core membership of the associated labeled bipartition are shown in columns 2 and 3 of Table 1.

Each of these matrices was also subjected to the comparable *UCINET* procedure. As is suggested, several starting configurations were used for each matrix. The highest correlation and the core membership of its associated labeled bipartition are shown in

[3] In addition, this approach allows us the necessary empirical context to test for the type of core/periphery structure that our data exhibit.

columns 4 and 5 of Table 1. Column 6 of Table 1 compares the results from the *UCINET* (Version 6.59) algorithm to the exhaustive search by showing the rank out of the 238 possible fitness values that was actually found by *UCINET*. For example, a rank of 1 indicates that *UCINET* found the global optimum from at least one of several starting configurations for a given matrix. Similarly, a rank of 5 indicates that *UCINET*'s best result was fifth best compared with the exhaustive list of possible nontrivial fitness values.

To increase the comparative value of this exercise, we selected three additional optimization search algorithms that might also have general applicability. For the first two algorithms, we used two of *Mathematica*'s built-in algorithms: a genetic algorithm, referred to there as "differential evolution," and a simulated annealing algorithm to optimize our fitness criterion. Finally, we programmed in *Mathematica* our own version of the Kernighan-Lin (1970) algorithm (see the Appendix), again using the same fitness criterion. For all 12 groups, all three of these algorithms matched the exhaustive search by consistently finding the global optimum from several starting configurations.

[Table 1 about here]

From the results in Table 1, the genetic algorithm in *UCINET* finds the global optimum in two out of our 12 cases. The *UCINET* fit statistic is among the five best for seven of the 12 cases, and among the ten best for nine of the 12 cases. All 12 solutions are in the top 20 out of the 238 possible solutions.

While it is next to impossible to debug a program without looking at the code, a comparison of the results from *UCINET* and the exhaustive searches does suggest a possible pattern. In the ten cases where the two solutions are different, the difference in the mean of the core block values and the mean of the periphery block values is larger for nine of the *UCINET* labeled bipartitions, and is the same for one case. These findings are more consistent with the analogy used by Borgatti and Everett (1999:384) when discussing the application of the discrete core/periphery model to valued data. Although maximizing this difference is a potentially defensible and interpretable fit criterion, it is not the stated fitness criterion in either their article or in *UCINET*.

4. Comparing algorithms where global optimum is unknown

Considering our results for relatively small data sets, it would also be useful to compare these same algorithms using somewhat larger data sets to see if there are any differences in this context. We made this comparison with the same data used by Borgatti and Everett (1999) to demonstrate the detection of discrete core/periphery structures in data. Baker's (1992) data reflect the number of citations from one journal to another among 20 social work journals over a 1-year period between 1985 and 1986. Given the computational requirements ($2^{20}-42 = 1{,}048{,}534$ possible nontrivial core/periphery bipartitions), an exhaustive search was not carried out. Instead, we use the results from *UCINET* as a benchmark. Table 2 compares the algorithms using the asymmetric dichotomous matrix (Borgatti and Everett 1999:385 Table 7) and the valued symmetric matrix (Borgatti and Everett 1999:386 Table 8)[4] of the Baker (1992) data. Consistent with Borgatti and Everett's (1999) analyses, diagonals for both matrices are ignored.

[4] Borgatti and Everett (1999:385) describe this as the "raw" citation data. However, an inspection of the table shows that it was actually symmetrized by choosing the larger of a_{ij} and a_{ji}, with two errors (SW-BJSW and ASW-JSWE cells).

[Table 2 about here]

The results shown for both *UCINET* and the simulated annealing algorithm represent the best fitness obtained from ten different starting configurations, which did vary for both algorithms. The alternative genetic algorithm (differential evolution) and the Lin-Kernighan algorithm match, but do not improve upon, the best *UCINET* genetic algorithm results, but they both consistently produce the same results from multiple starting configurations. For these data, Table 2 shows that the simulated annealing algorithm, as implemented in *Mathematica*, does not perform nearly as well as the others.

Taken together, these two illustrations suggest that the *UCINET* implementation of the genetic algorithm and *Mathematica*'s version of the simulated annealing algorithm are not the best choices for consistently finding global optima for discrete core/periphery bipartitions of both small and larger sized data sets. It would be unwise to develop a generic permutation test for fitness values from these two algorithms because of the increased likelihood of Type II errors. *Mathematica*'s version of the genetic algorithm and our implementation of the Lin-Kernighan algorithm are both superior choices for this purpose. To select the better of these two choices, we considered relative computing time.

The best core for the 20 by 20 Baker (1992) asymmetric binary data, found by both the Kernighan-Lin (in 3.0 seconds[5]) and differential evolution (35 seconds) algorithms, was {CW, JSWE, SCW, SSR, SW, SWRA}, with fitness 0.82561. Unfortunately, simulated annealing (in 43 seconds) found a suboptimal core, with a fitness of only 0.63457, that differs from the previous core by the addition of the journals CYSR, JSP, and SWG. Note that if any one of these three added journals is switched to the periphery, there is a gain in fitness, showing that the simulated annealing solution is not even a local optimum. Finally, the Kernighan-Lin algorithm has been estimated to run in $O(n \log n)$ time for the graph partition problem (Fiduccia and Mattheyses, 1982). Conclusion: the Kernighan-Lin solution is tied with differential evolution for the quality of its solutions, but is much faster, which is important when doing a permutation test requiring 1000 or more repetitions of the algorithm.

5. A generic permutation test for discrete core/periphery fitness values

Now that we are fairly confident that we have a reasonably fast search algorithm that finds, or at least closely approximates, the global optimum, we would now like to attach a significance or probability value to our observed fitness value. Again, our objective is to develop a permutation test that will enable us to ascertain the likelihood of obtaining by chance alone a fitness value as high as our observed fitness value. Ideally, we would like this permutation test to be applicable for all types of data that could conceivably be tested for evidence of a discrete core/periphery structure (e.g., binary or valued, symmetric or directed)

To do this, we would need to effectively reverse the argument as stated by Borgatti and Everett (1999:393–394). We require a method of permutation that is independent of the context of our data so that any assumptions about how ties form are minimized. Additionally, this generalized permutation test still needs to be specific enough to a particular dataset to allow for a meaningful and interpretable comparison of an observed fitness value to a permutation baseline.

[5] Timed for a Dell Inspiron 8200 with a 1.7 GHz Pentium 4. The programs were not optimized.

A reasonable solution here is to preserve and focus upon permutations of the cell distribution of the matrix (or the edge distribution of the graph), which also preserves the total number of ties. However, row and column marginals are not preserved. For each permutation of the cell distribution, we also want to obtain its optimal core/periphery bipartition and record the fitness value. By generating a large number of these permutations and recording the fitness values, we can compare our observed fitness value to this distribution of permutation fitness values to calculate a likelihood of observing by chance alone a fitness value equal to or higher than our observed fitness value. We can then use this probability in combination with our observed fitness value to make a more definitive assessment of the degree to which data exhibit a core/periphery structure.

The p-value from the permutation test is defined as the proportion of fitness values f_i from the random permutations that are greater than or equal to the observed fitness f_0 (on the original data). A high p-value, say greater than .05, suggests that the observed data may not be adequately described by a core/periphery structure. By contrast, a low p-value indicates that only a very small proportion, namely p, of the random permutation fitness values are at least as strong as the observed fitness. A low probability along with an intuitively high observed fitness value suggests that the observed data may have a core/periphery structure.

To illustrate this permutation test, we used *Mathematica* to program a random permutation generator based upon the observed within group distribution of messages for each of the 12 groups from Table 1. As with the observed data, diagonal cells were also ignored for these permutations. For all 1,000 random permutations for each group, we also obtained the optimal core/periphery fitness values using our exhaustive search algorithm.

The results of three independent runs of this method for each group are shown in Table 3. The three p-values for each group represent the proportion of permuted optimal fitness values (out of 1,000) that were equal to or greater than the observed fitness value. For Group 1, for example, no random permutation in each of the 3 runs produced an optimal fitness value equal to or greater than the observed fitness value of 0.867 (see Table 1). For Group 3, 43 of the random permutations in the first run produced optimal fitness values equal to or greater than the observed fitness value (0.650). There were 44 such permutations in the second run, and 41 permutations in the third run.

[Table 3 about here]

It is also useful to be able to inspect the distribution of permutation fitness values and visually compare this with the observed fitness values. A standard way to do this is to plot a cumulative distribution of permutation fitness values, ordered from lowest to highest, and to place the observed fitness value where it falls in this distribution. This plot is shown for Group 1 in Fig. 1, and for Group 7 in Fig. 2 of the Beck et al.(2003; ND) data. Both Figures contain the permutation fitness values from Run 3 (Table 3). In Fig. 1 the optimal fitness, $f_0 = 0.867$, is well above all the fitness values for the random permutations of the data, which range from 0.235, the leftmost value, to 0.801, the rightmost. All permutation fitness values were found by exhaustive search, so we are confident that they are correct.

[Fig. 1 about here]

[Fig. 2 about here]

For comparative purposes, the best fitness value obtained using *UCINET* is represented in the plot by a horizontal line at f_U. For Group 1 the highest fitness that *UCINET* could find, $f_U = 0.570$, crosses the random fitness curve at about the 80% level, as indicated in Fig.1. All 1000 of the fitness values for the random permutations are less than that of the original data. This allows us to conclude that the *p*-value for f_0 is less than 0.1%, and that the core/periphery model is supported for Group 1.

Fig. 2 also plots fitness values from Beck et al.'s (2003; ND) communication data, but this time for Group 7. The optimal fitness is $f_0 = 0.565$, while the *UCINET* fitness, $f_U = 0.398$, is again suboptimal. The first ranked position where the permutation fitness values exceed f_0 is at 830. This means that the *p*-value is 0.170, which is not significant at the conventional 5% level. We therefore conclude that the structure of the overall communication data for Group 7 is not consistent with a core/periphery model.

Ties in permutation fitness values are indicated in both figures by a vertical line with a gap in the middle, and their low density, especially in the critical 95% range, indicates that they will not be a problem in interpreting the results. Table 4 shows the distribution of the length of runs of ties from the third run of Table 3 for each of the 12 data sets. Note that the maximum run length is five, again showing that long runs of identical fitness values is not a problem.

Most of the permutation fitness ties in Table 4 represent different cores that happen to have the same fitness values. By the "same" fitness values, we mean that the values are the same up to round-off error, corresponding to the *Mathematica* option, SameTest→Equal. To get the total run length from Table 4, sum the product of the j^{th} run-length by its frequency in the totals row. Therefore, the total run-length of repeated fitness values is 1058 plus 169×2, and so on, which equals 1476. However, 171 of the 1476 repeated fitness value runs (of the 12,000 total random permutations) could be distinguished by the size of the core, as determined from another *Mathematica* calculation. I.e., 171 ties could be broken by considering core length. Next, 1380 of these repeated fitness values could be distinguished by considering the two ordered vectors of values in the permuted core and periphery, respectively, leaving only 96 unbroken ties. E.g., in the twelfth data set, one of the pairs of vectors of randomized and then optimized core/periphery entries was, after ordering, (3,4,4,4,5,6) and (0,0,0,0,0,1,1,2,2,2,2,2,2,3,3,3,3,3,4,4). Another such random permutation led to (2,4,4,4,6,6) and (0,0,0,1,1,1,1,1,2,2,2,2,2,2,3,3,3,3,4,4). Note that both of these structures had the same core length, viz., three. The reader can verify that when these vectors are correlated with the image vector (six 1s followed by twenty 0s) the result in both cases is $149/\sqrt{53310} \approx 0.64533$. Finally, even the presence of these pairs of identical permutation fitness values and their respective core/periphery vectors does not indicate that the corresponding random permutations were identical, since there are many permutations that give rise to the same core/periphery ordered vectors.

[Table 4 about here]

The reader should note that all the fitness values on the randomly permuted data in both figures are positive. Since our fitness is just a correlation, restricted to a subset of the matrix, one might expect that the average fitness would be zero. However, since we

select the core/periphery bipartition that maximizes fitness, this number is always positive. This means that the usual interpretation of squared correlation as the "percent of variance explained" does not apply here. Therefore, the *p*-value is the only guide we can give as to whether a particular fitness is "large" or not.

Finally, Fig. 3 plots a cumulative distribution of 1000 permutation fitness values for the Baker (1992) journal citation data, in the asymmetric binary form from Borgatti and Everett (1999: 385 Table 7). This plot is analogous to the plots for Figures 1 and 2, but the ties are not indicated. The most notable difference is that the fitness values from the Kernighan-Lin, differential evolution, and UCINET are in agreement and are far greater than all the permuted values, although the running time for the former program was much faster. However, the simulated annealing program gave suboptimal answers for this problem. The fact that the fitness found by the Kernighan-Lin algorithm, 0.826, was so much higher than the maximum value, 0.441, of the 1000 random permutations shows that the core/periphery structure may be even more significant than is indicated by the *p*-value of less than 0.1%.

[Fig. 3 about here]

6. Discussion and conclusion

Our permutation test is very general in that it can detect any deviation from randomness in the fitness measure. One might argue that we should have controlled for symmetry, marginal totals, or other network variables. However, symmetry is not specified in the core/periphery model, and it is conceptually hard to separate highly variable and correlated marginals from the definition of a core/periphery structure. For instance, if the within core ties are dense, while the within periphery ties are sparse, then both core marginals tend to be relatively large, while both periphery marginals would be smaller. Not controlling for plausible variables increases the danger of Type I errors (falsely rejecting the null hypothesis). One suspects Type I errors when too many tests seem to be significant. It is true that we found that all of the first four Beck et al. (2003; ND) data sets had a highly significant core/periphery structure, but of the remaining eight groups (Beck et al., ND) three or four (groups 5, 7, 11, and possibly 8) are not significant at the 5% level. This is in general agreement with the observations in their article. This suggests that our test is powerful enough to detect core/periphery structures when they exist, but not so generous as to allow almost anything to be a core/periphery structure.

However, it is easy to modify our permutation test for perfectly symmetric data: one permutes and correlates only the upper diagonal entries. Of course, one could choose the lower diagonal entries just as well. In fact, one could use any fixed random choice of one entry from each pair, A_{ij} and A_{ji}, for all unordered pairs $\{i, j\}$.

This last remark suggests how to control for a probabilistic symmetry bias, where A_{ij} and A_{ji} are not independent. The exact form of the dependence may be very complicated, especially for valued data, but none of that matters with the following permutation scheme. Instead of permuting single entries, $A_{i,j} \alpha A_{\sigma(i,j)}$, determined by a diagonal-fixing permutation σ of ordered pairs (i, j), one permutes symmetric *pairs* of entries, determined by any permutation τ defined on the set of all unordered pairs $\{i, j\}$ of vertices. Any such permutation preserves the exact symmetry bias, expressed as the conditional probability, $P(A_{ji} = y \mid A_{ij} = x)$.

One could also propose similar ideas for bringing in the diagonal elements by using permutations that permute both diagonal and off-diagonal elements, with the restriction that diagonal elements are permuted among themselves, and the same for off-diagonal elements. The fitness measure would also be a correlation, but this time extended to include the diagonal elements. However, the authors strongly believe that this extension would violate the intuitive idea that the core/periphery model is exclusively about relations between different elements, be they individuals or countries, not properties that these elements possess by themselves.

Fitness measures other than the Pearson product moment correlation can, of course, be used within the scheme of this paper. One merely substitutes the new fitness function in the optimization program. Examples of alternative fitness measures would be a simple matching coefficient, a chi-square measure, or a log-likelihood function. The matching coefficient might improve the speed of the calculations, but it fails to control for the size of the two groups. The latter two measures are likely to be highly correlated with the correlation coefficient. Two of the *UCINET* options, DENSITY (the density of the core block interactions) and SXY (the element wise product of the permuted data matrix and an ideal structure matrix consisting of ones in the core block interactions and zeros in the peripheral block interactions) may also be correlated with the Pearson product moment correlation. However, the *UCINET* option EMPTYPER (the number of entries in the peripheral block interactions) may give quite different results.

A potential problem with permutation tests is found in highly skewed data, as shown by Faust and Romney (1985) for network data with coefficients of skewness γ_1 over 2.8. The skewness of the combined Beck et al. (2003; ND) data (excluding the undefined diagonal values) is 0.868. While this is statistically significant (9.2 standard deviations on the positive side of 0), it is not as excessive as in the Faust and Romney study (1985). If one attempts to correct our skewness by the widely used $\log(1+x)$ transformation, one "over-shoots" and the skewness is now negative, at –0.318. Using trial and error, we found that the transformation $x^{2/3}$ largely eliminates the skewness in the Beck et al. (2003, ND) data at 0.049. However, it is difficult to justify, either theoretically or substantively, such empirically derived transformations.

Although the Kernighan-Lin program was faster than the others considered here, it could be greatly speeded up by several simple changes. The main improvement is that the gain (or loss) of fitness resulting from switching the core/periphery membership of a single point could be efficiently computed, instead of the easy, but slow, way it was actually implemented: recomputing the fitness of the new structure from scratch and subtracting it from the old fitness. This improvement would give the same proportional increase in speed for the differential evolution and simulated annealing programs, as they had the same gain function. Other improvements would involve reprogramming crucial parts of the algorithm in C and linking these with the main *Mathematica* program. With these changes, much larger data sets could be analyzed.

The Beck et al. (2003; ND) data was ideally suited to explore the algorithms considered here (*UCINET*, Differential Evolution, Simulated Annealing, and Kernighan-Lin) because of the small size (twelve 8 by 8 matrices) that allowed us to compare them all with the gold-standard, exhaustive search. *UCINET* was found to give suboptimal answers. As implemented in *Mathematica*, the differential evolution algorithm gave correct answers, but was slow. The Kernighan-Lin algorithm emerged as the best on both

counts, accuracy and speed, and has the potential to be useful in finding and testing core/periphery structures.

References

Aarts, E., Lenstra, J. K., 1997. Introduction. In: Aarts, E., Lenstra, J. K. (Eds.), Local Search in Combinatorial Optimization. Wiley, Chichester, pp. 1–18.

Alba, R.D., Moore, G., 1978. Elite social circles. Sociological Methods and Research 7, 167–188.

Baker, D. R., 1992. A structural analysis of the social work journal network: 1985–1986. Journal of Social Service Research 15, 153–168.

Barsky, Noah P., 1999. A core/periphery structure in a corporate budgeting process. Connections 22(2), 22–29.

Beck, R. J., Fitzgerald, W. J., Jester, R. B., ND. The effects of levels of instruction on core-periphery formation and communications quality in online discussion groups. Under Review for Small Group Research #04028.

Beck, R. J., Fitzgerald, W. J., Pauksztat, B., 2003. Individual behaviors and social structure in the development of communication networks of self-organizing online discussion groups. In: Wasson, B., Ludvigson, S., Hoppe, U. (Eds.), Designing for Change in Networked Learning Environments. Kluwer, Dordrecht, pp. 313–322.

Borgatti, S. P., Everett, M. G., 1999. Models of core/periphery structures. Social Networks 21, 375–395.

Borgatti, S. P., Everett, M. G., Freeman, L. C., 2002. *UCINET* for Windows, Version 6.59: Software for social network analysis. Analytic Technologies, Harvard.

Boyd, J.P., 2002. Finding and testing regular equivalence. Social Networks 24,315–331.

Boyd, J.P., Jonas, K.J., 2001. Are social equivalences ever regular? Permutation and exact tests. Social Networks 23(2), 87–123.

Cummings, J. N., Cross, R., 2003. Structural properties of work groups and their consequences for performance. Social Networks 25(3), 197–210.

Doreian, P., 1985. Structural equivalence in a psychology journal network. American Society for Information Science 36(6), 411–417.

Faust, K., Romney, A.K., 1985. The effect of skewed distributions on matrix permutation tests. British Journal of Mathematical and Statistical Psychology 38, 152–160.

Fiduccia, C.M., Mattheyses, R.M., 1982. A linear-time heuristic for improving network partitions. In: ACM IEEE Nineteenth Design Automation Conference: Proceedings, IEEE Computer Society Press, Los Alamitos, CA, pp. 175–181.

Goldberg, D. E., 1989. Genetic Algorithms. Addison Wesley, New York.

Good, P., 1994. Permutation Tests. Springer-Verlag, New York.

Kernighan, B. W., Lin, S., 1970. An efficient heuristic procedure for partitioning graphs. Bell System Technical Journal 49, 221–226.

Kirkpatrick, S., Gelatt, Jr., C.D., Vecchi, M. P., 1983. Optimization by simulated annealing. Science 220, 671–680.

Laumann, E.O., Pappi, F.U., 1976. Networks of Collective Action: A Perspective on Community Influence Systems. Academic Press, New York.

Mintz, B., Schwartz, M., 1981. Interlocking directorates and interest group formation. American Sociological Review 46, 851–868.

Smith, D., White, D., 1992. Structure and dynamics of the global economy: network analysis of international trade 1965–1980. Social Forces 80, 857–893.

Wolfram, S., 2003. The Mathematica Book, 5th ed. Wolfram Media.

Table 1

Comparison of core/periphery fitness measures using Beck et al. (2003; ND) data

	Exhaustive Search*		***UCINET* Genetic Algorithm**		
Group #	**Fitness**	**Core Membership**	**Fitness**	**Core Membership**	**Rank**
1	0.867	C,D,E	0.570	A,B,C,D,E	19
2	0.834	A,D,F,G	0.672	B,D,E,F,G	10
3	0.650	D,E,F,H	0.650	D,E,F,H	1
4	0.875	B,F	0.737	B,C,F	2
5	0.559	A,B,E,G,H	0.467	A,B,E,F,G,H	10
6	0.656	C,E,H	0.444	C,D,E,G,H	19
7	0.565	A,C,E,F,G	0.398	A,C,D,E,H	18
8	0.625	A,B,D	0.501	A,B,C,D,G	4
9	0.728	B,G,H	0.689	B,E,G,H	3
10	0.645	A,D,H	0.645	A,D,H	1
11	0.535	C,E,F,G	0.471	B,C,E,F,G	2
12	0.672	C,D,F,G	0.529	C,D,F,G,H	5

*Same results for alternative genetic, simulated annealing, and Kernighan-Lin algorithms

Table 2

Comparison of core/periphery fitness measures using Baker (1992) data

	***UCINET* Genetic Algorithm ***		**Simulated Annealing Algorithm**	
Data Type	**Fitness**	**Core Membership**	**Fitness**	**Core Membership**
Asymmetric, Dichotomous	0.826	CW, JSWE, SCW, SSR, SW, SWRA	0.712	BJSW, CW, CYSR, JSWE, PW, SCW, SSR, SW
Valued, Symmetric	0.815	SCW, SSR, SW	0.575	ASW, CYSR, CSWJ, SSR, SW

*****Same results for alternative genetic and Kernighan-Lin algorithms

Table 3

Permutation tests for Beck et al. (2003; ND) data

Group #	Run 1*	Run 2*	Run 3*
1	0.000	0.000	0.000
2	0.000	0.000	0.000
3	0.043	0.044	0.041
4	0.000	0.000	0.001
5	0.150	0.184	0.174
6	0.035	0.034	0.034
7	0.134	0.169	0.170
8	0.053	0.049	0.044
9	0.006	0.011	0.004
10	0.048	0.036	0.046
11	0.271	0.282	0.279
12	0.022	0.015	0.016

*N=1000

Table 4. The distribution of run length of permutation fitness values for Beck et al. (2003; ND) Data**.**

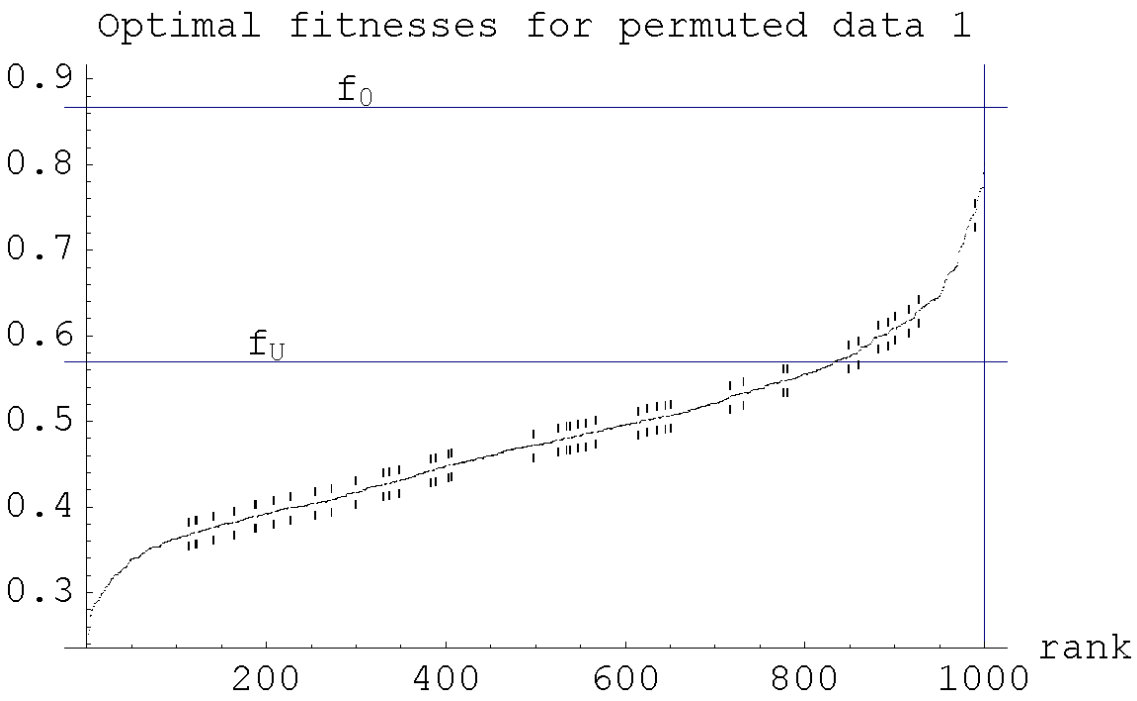
Group #	**0**	**1**	**2**	**3**	**4**	**5**
1	915	38	3	0	0	0
2	839	68	7	1	0	0
3	801	81	11	1	0	0
4	859	60	3	3	0	0
5	679	121	25	1	0	0
6	641	119	33	4	0	1
7	769	89	11	5	0	0
8	653	132	22	3	1	0
9	864	65	2	0	0	0
10	715	102	21	3	0	1
11	748	97	18	1	0	0
12	792	84	10	1	0	1
Totals	9276	1058	169	27	6	9

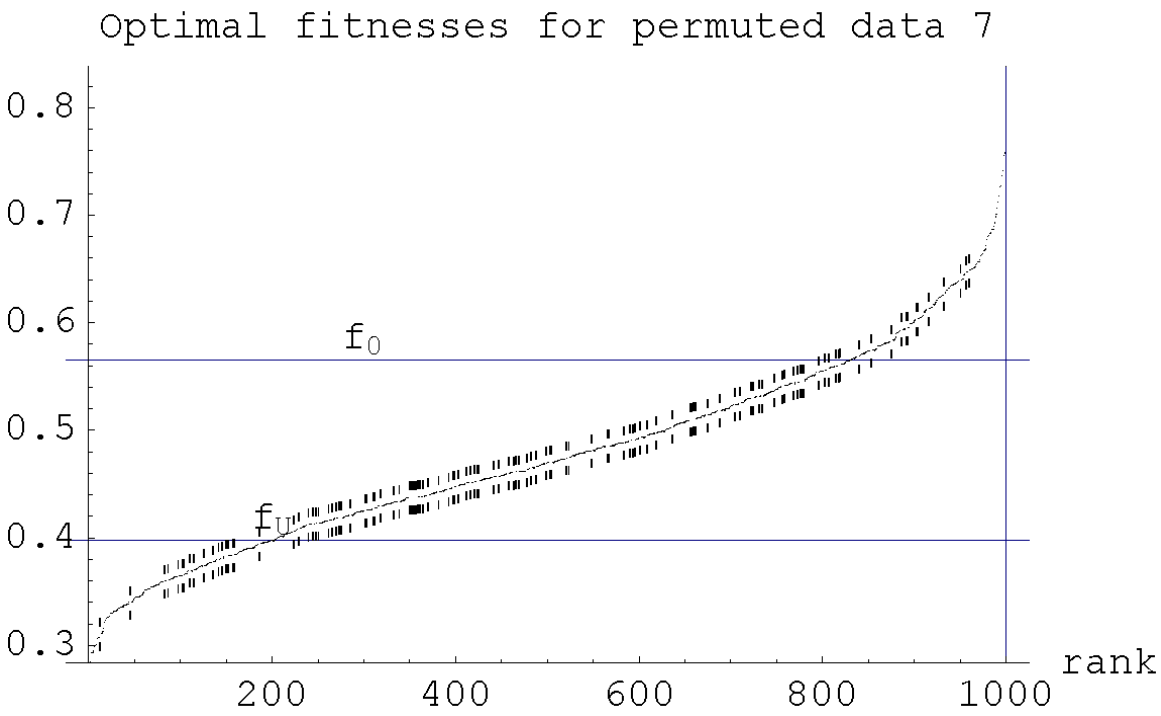
Captions for Figures

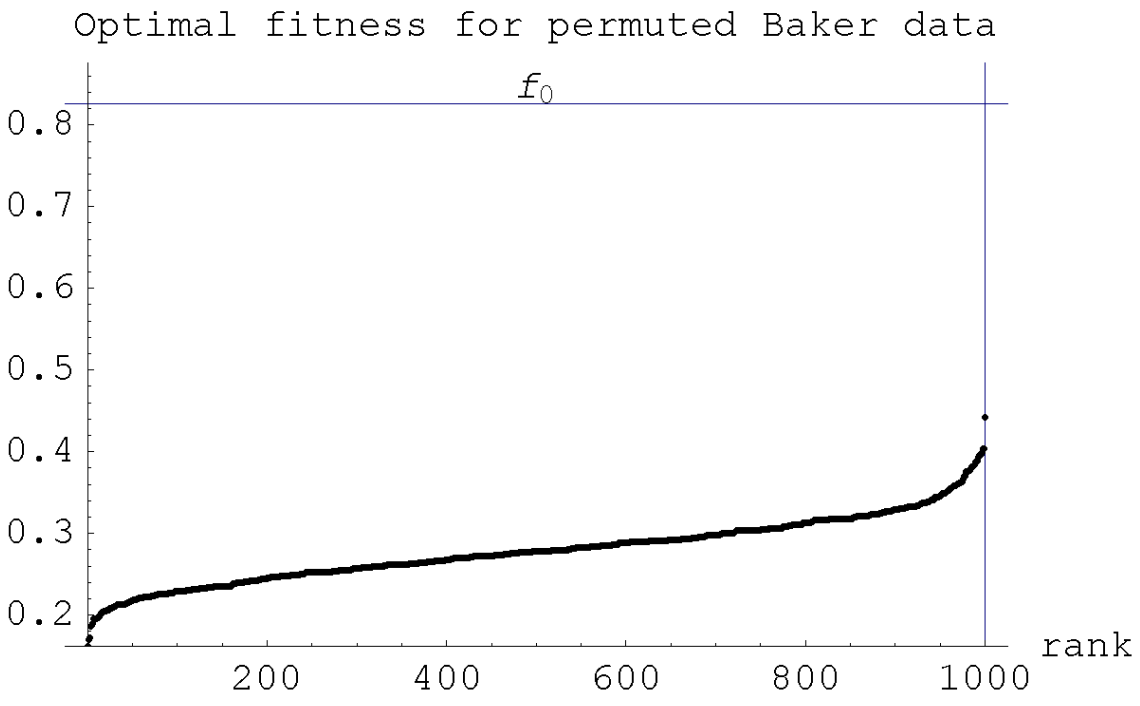
Fig. 1. Fitness values of permutations of Group 1 from Beck et al.'s (2003; ND) data.

Fig. 2. Fitness values of permutations of Group 7 from Beck et al.'s (2003; ND) data.

Fig. 3. Fitness values of permutations of the asymmetric binary Baker (1992) journal citation data.


Optimal fitnesses for permuted data 1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
f_0
f_U
200
400
600
800
1000
rank


Optimal fitnesses for permuted data 7
0.8
0.7
0.6
0.5
0.4
0.3
f_0
f_U
200
400
600
800
1000
rank


Optimal fitness for permuted Baker data
f_0
0.8
0.7
0.6
0.5
0.4
0.3
0.2
200
400
600
800
1000
rank

Appendix: a Kernighan-Lin algorithm for the core/periphery problem

The gain function $g(a)$, represents the fitness gain from moving a, one of the n individuals, from its current block (core or periphery) to the other block. The outer loop (**Step 4**) is typically repeated only two or three times.

Step 0. Pick a random core/periphery bipartition with at least individuals in each block.
Step 1. Choose an individual a such that $g(a)$ is maximal (even if not positive).
Step 2. Perform a "tentative" reassignment of a to the other block.
Step 3. Repeat **Steps 1** and **2** exactly n times, where an individual cannot be chosen to be reassigned (the "locking rule") if she has already been reassigned in one of the previous iterations of **Steps 1** and **2**, but within the current loop of **Step 4**. This sequence of reassignments defines a sequence of gains $g_1,\mathrm{K}\ ,g_n$, where g_i corresponds to the reassignment of individual a_i to the other block in the i^{th} iteration. The total gain after k reassignments equals $G(k) = \sum_{i=1}^{k} g_i$. N. b., for $k = n$, every individual has been reassigned to a different block, and $G(n) = -f_0$, where f_0 is the fitness at the beginning of the current **Step 4** loop.
Step 4. Choose the value of k for which $G(k)$ is maximal. If $G(k) > 0$, the local neighborhood solution is given by the partition obtained from the initial partition by the "definite" reassignment of $(a_1,\mathrm{K}\ ,a_k)$, and **Steps 1** through **4** are repeated (resetting the "locks"). If $G(k) \le 0$, then we are done.