J. Math. Biol. (2007) 54:199–226
DOI 10.1007/s00285-006-0048-4

Mathematical Biology

An algorithm for a decomposition of weighted digraphs: with applications to life cycle analysis in ecology

L. Sun · M. Wang

Received: 18 August 2003 / Revised: 22 March 2006 /
Published online: 8 November 2006

Abstract In the analysis of organism life cycles in ecology, comparisons of life cycles between species or between different types of life cycles within species are frequently conducted. In matrix population models, partitioning of the elasticity matrix is used to quantify the separate contributions of different life cycles to the population growth rate. Such partition is equivalent to a decomposition of the life cycle graph of the population. A graph theoretic spanning tree method to carry out the decomposition was formalized by Wardle [Ecology **79**(7), 2539–2549 (1998)]. However there are difficulties in realizing a suitable decomposition for complex life histories using the spanning-tree method. One of the problems is the occurrence of life cycles that contain contradictory directions that defy biological interpretation. We propose an algorithmic approach for decomposing a directed, weighted graph. The graph is to be decomposed into two parts. The first part is a set of simple cycles that contain no contradictory directions and that consist of edges of equal weight. The second part of the decomposition is a subgraph in which no such simple cycles are obtainable. When applied to life cycle analysis in ecology, the proposed method will

Although the research described in this article has been funded in part by the United States Environmental Protection Agency through STAR cooperative agreement R-82940201-0 to the University of Chicago, it has not been subjected to the Agency's required peer and policy review and therefore does not necessarily reflect the views of the Agency and no official endorsement should be inferred.

L. Sun
Department of Mathematics, Beijing Institute of Technology, Beijing, China

M. Wang (✉)
Department of Statistics, The University of Chicago, 5734 S. University Ave.,
Eckhart Hall, Room 106, Chicago, IL 60637, USA
e-mail: meiwang@galton.uchicago.edu

guarantee a complete decomposition of the life cycle graph into individual life cycles containing no contradictory directions.

Keywords Weighted digraph · Graph decomposition algorithm · Spanning tree · Life cycle analysis · Loop analysis

1 Introduction

The goal of this study is to develop a graph theoretic algorithmic approach that decomposes weighted digraphs and provides a useful method in life cycle analysis in population ecology. To evaluate the importance of different life histories, comparisons of life cycles between species or between different types of life cycles within species are frequently conducted. This comparison is particularly useful in the study of trade-offs between different components of fitness and reproduction or in evaluating distinct history tactics among individuals. Matrix population models have been used widely for modeling biological populations and the analysis of life cycles [3]. A population projection matrix is essentially a type of transition matrix for Markov processes (see the description in Sect. 4.1). The dominant eigenvalue λ of the population projection matrix gives the long-term growth rate of the population. The corresponding elasticity matrix consists of elements that represent the proportional sensitivity of λ with respect to the elements in the population projection matrix. The total elasticity (i.e. the sum of all elements of the elasticity matrix) is 1, or 100%. Partitioning of the elasticity matrix is used to quantify the separate contributions of different life cycles to the population growth rate [6]. This method is called *loop analysis*, because the components of the decomposition correspond to loops in the life cycle graph.

A life cycle graph is a graphical description of the life cycles of a population. The nodes (*a.k.a.* vertices, points) of the graph represent the life stages, the directed line (*a.k.a.* arc, edge) from node j to node i indicates that an individual in stage j at time t can contribute individuals to stage i at time $t+1$, i.e., the (i,j)th element in the population matrix is not zero. Therefore, there is a correspondence between a population projection matrix of n life stages and a life cycle graph of n nodes, where directed arcs connecting the nodes correspond to non-zero elements in the projection matrix (see examples in Sect. 4.4). Furthermore, each directed arc can be assigned a weight. In loop analysis, the weight for the directed arc from node j to node i is the (i,j)th element in the elasticity matrix. The life cycle graph of the population can be decomposed into a set of *loops* (closed paths or cycles) that correspond to life history pathways followed by individuals in the population. Each loop is given equal weights of elasticities on every arc in the loop (details given in Sects. 4.3 and 4.4). The sum of the weights of all loops is 100% – the sum of all elasticities. There is a biological explanation for the decomposition: the decomposition illustrates and quantifies the contributions of different life cycles of individuals to the growth rate of the population [3,6,13].

The decomposition of life cycle graphs can be done by inspection if the number of nodes (life stages) is small ($\leq$4). For general purposes, a spanning

tree method was summarized and illustrated by Wardle [13]. The spanning tree method is a systematic approach. Starting with a base tree that has all n nodes connected by $n-1$ edges and contains no loops, loops are formed by adding edges to the base tree one at a time. The spanning tree method will produce a set of independent cycles (see Example 4.1).

In practice, using the spanning tree method may lead to two problems. First, for moderately complicated graphs, it is often hard or impossible to find a tree that spans a set of cycles containing no contradictory directions. It is clear that cycles containing contradictory directions do not represent life cycles of individual organisms, thus defying biological interpretation [13]. Second, each tree spans a fixed set of cycles; a pair of cycles that may be interesting for comparison purposes might not show up in the same set of cycles. One could modify the situation by combining nodes to reduce the complexity of the graph, but the modification is often limited and not satisfactory.

We propose an algorithmic approach to decompose connected, non-negatively weighted directed graphs. Applying our method to loop analysis in population ecology, one obtains a set of independent cycles containing no contradictory directions. The decomposition is not unique. In applications such as life cycle analysis, important, meaningful cycles can be given higher priority to be selected by the algorithm.

Section 2 provides the graph theory terminology, the proposed algorithm and the proofs of corollaries relevant to the algorithm. Section 3 uses three simple, hypothetical examples to illustrate the algorithm. Section 4 discusses the applications to life cycle analysis, with two examples of complex life cycle graphs. Conclusions are discussed in Sect. 5. Details of the matrix decompositions of the two examples in Sect. 4 are given in the Appendix.

2 The algorithmic approach

2.1 Terminology

A *directed graph* or *digraph* G can be described as $G = (V, E)$, where $V = \{v_1, v_2, \ldots, v_n\}$ is the set of *vertices* or *nodes* of the graph G, $E = \{e_{ij} = (v_i, v_j), 1 \le i, j \le n\}$ is the set of *directed edges* or *arcs* of G that consists of ordered pairs of vertices of G. $|E(G)|$ denotes the total number of edges of the graph G. A *weighted digraph* $(G, \mathbf{w})$ is a digraph G with a numerical weight assigned to each directed edge. Such weighted digraph has a matrix representation (see illustrative examples in Sect. 3):

$$
\begin{array}{c} \\ \text{to} \end{array}
\begin{array}{c}
\begin{array}{cccc} \text{from vertices} & & & \\ v_1 & v_2 & \cdots & v_n \end{array} \\
\begin{bmatrix}
w_{11} & w_{12} & \cdots & w_{1n} \\
w_{21} & w_{22} & \cdots & w_{2n} \\
\vdots & \vdots & \vdots & \vdots \\
w_{n1} & w_{n2} & \cdots & w_{nn}
\end{bmatrix}
\end{array}
\begin{array}{l}
\\
v_1 \\ v_2 \\ \vdots \\ v_n
\end{array}
$$

where w_{ij} is the weight of the directed edge from vertex v_j to vertex v_i. Notice that the transpose of the above matrix might be more customary in some mathematical fields. The notation used here is more conventional in applications in population biology.

A *cycle* (according to [1,7]) or a *simple cycle* (according to Carré [2]) is a closed trajectory of a sequence of edges in G that connects nodes $v_{i_1}, v_{i_2}, \ldots, v_{i_k}, v_{i_1}$ consecutively, where $v_{i_1}, \ldots, v_{i_k}$ are distinct. A graph G is called *connected* if any two of its vertices are linked by connected edges, i.e. by a *path* [5] in G.

In this paper, we consider connected digraphs with $n < \infty$ vertices and nonnegative weights ($w_{ij} \geq 0$, for all $i, j = 1, 2, \ldots, n$) on all edges. We develop an algorithm that decomposes a weighted digraph into a set of simple cycles, which contain no contradictory directions and are of equal weights on every edge in each cycle, and a remaining subgraph on which no such cycles are obtainable.

2.2 The algorithm

The decomposition method can be described by the following steps:

1. Start with a connected digraph G with $|E(G)|$ directed, weighted edges.
2. Consider a possible simple cycle that starts and ends at a specific node. Extract a simple cycle without contradictory directions from G by searching among all possible simple cycles without contradictory directions with the given ending node.
   This step can be illustrated by the following search for a simple cycle that starts and ends at a node, say, v_1, under the condition that weight $w_{i,1} > 0$ for some $i \neq 1$ (i.e. there is at least one edge from v_1), and $w_{1,j} > 0$ for some $j \neq 1$ (there is at least one edge going to v_1).
   (a) Search through the rest nodes $v_2, \ldots, v_n$. Choose the first i_1 such that $w_{i_1,1} > 0$ (edge $v_1 \to v_{i_1}$ exists).
   (b) If $w_{1,i_1} > 0$ (edge $v_{i_1} \to v_1$ exists), the selection is completed: the cycle selected is $v_1 \to v_{i_1} \to v_1$. Else search through the rest nodes $\{v_2, \ldots, v_n\} \setminus v_{i_1}$ (i.e. all nodes but v_{i_1}). Choose the first i_2 such that $w_{i_2,i_1} > 0$. (If $w_{j,i_1} = 0, \forall j$ (i.e. for all j), go back to Step 1 to choose another i_1 such that $w_{i_1,1} > 0$, then proceed.)
   (c) If $w_{1,i_2} > 0$, the selection is completed: the cycle selected is $v_1 \to v_{i_1} \to v_{i_2} \to v_1$. Else repeat step 2: search through $\{v_2, \ldots, v_n\} \setminus \{v_{i_1}, v_{i_2}\}$, choose the first i_3 such that $w_{i_3,i_2} > 0, \ldots$ and see if $w_{1,i_3} > 0$, etc. (If $w_{j,i_2} = 0, \forall j$, go back to Step 2 to choose another i_2 such that $w_{i_2,i_1} > 0$, then proceed.)
   (d) If the cycle cannot be completed for all j such that $w_{j,1} > 0$, there are no simple cycles starting and ending at v_1.
3. Once a simple cycle L_1 is obtained, each edge will be assigned the same weight that is the least edge weight among all edges of L_1.
4. G has decomposed to $G = G_1 \cup L_1$, and $|E(G_1)| < |E(G)|$.

Here the union of weighted digraphs corresponds to the element-wise matrix addition of the weight matrices of the graphs:

$$\left(w_{ij}^{G}\right)_{n\times n} = \left(w_{ij}^{G_1}\right)_{n\times n} + \left(w_{ij}^{L_1}\right)_{n\times n}$$

where $w_{ij}^{G}, w_{ij}^{G_1}$ and $w_{ij}^{L_1}$ are the edge weights of graphs G, G_1 and L_1, respectively.

5. Repeat the above steps on G_1 and get a new simple cycle L_2 in G_1.
6. At the end of this procedure,

$$G = G_r \bigcup \left(\bigcup_{i=1}^{r} L_i\right)$$

where L_i's are simple cycles without contradictory directions, G_r is a subgraph of G with no such cycles obtainable.

2.3 Corollaries

There are two immediate corollaries on the properties of the decomposition. The corollaries are related to the concept of *flow conservation* described in the following definition.

Definition (The flow conservation condition) *A weighted digraph satisfies the Flow Conservation Condition if* $\forall i$,

$$\sum_{j=1}^{n} w_{ij} = \sum_{j=1}^{n} w_{ji} \tag{1}$$

where n is the number of nodes in G.

The above definition is equivalent to the definitions in Carré [2] and Jungnickel [7] with the point of view of networks and flows. If w_{ij} is treated as the amount of flow of some substance from node j to node i, then $\sum_j w_{ij}$ is the total amount flowing into node i, $\sum_j w_{ji}$ is the total amount flowing out of node i. The flow conservation condition asserts equal amounts of inflow and outflow at each node. The term "balanced" is used in [1] for individual vertices satisfying Condition (1), and the term "circulation" in [1] is essentially equivalent to the definition of the flow conservation condition under a more general setting that involves costs and path lengths.

Corollary 2.1 *If G satisfies the flow conservation condition, then the remainder graph* $G_r = \emptyset$ (*the empty set*).

Proof The idea behind the proof is straight forward. Each cycle in the decomposition satisfies the flow conservation condition. The flow conservation condition is additive. Thus the remainder graph G_r must also satisfy the condition, and since G_r contains no cycles, it must be empty. The details follow and may appear cumbersome with all the subscripts, superscripts and summations.

The corollary comes from the following two facts:

1. Any simple cycle L of G with equal edge weight satisfies the flow conservation condition (1) at every node $v_i \in G$.
   Assume that L has path $v_{i_1} \to v_{i_2} \to \cdots \to v_{i_k} \to v_{i_1}$, where $i_m, m = 1, 2, \ldots, k$, are distinct, and each edge of L is of equal weight w. Let w_{ij}^L denote the edge weight for the weighted digraph L. Then for any vertex $v_i \in G$,

$$\sum_j w_{ij}^L = \begin{cases} w_{i_m, i_{m-1}}^L = w = w_{i_{m+1}, i_m}^L = \sum_j w_{ji}^L & \text{if the vertex } v_i = v_{i_m} \in L \\ 0 = \sum_j w_{ji}^L & \text{if the vertex } v_i \notin L \end{cases}$$

   where we use the convention that $i_{m-1} = i_k$ if $i_m = i_1$, $i_{m+1} = i_1$ if $i_m = i_k$ and $w_{ij}^L = 0$ if the corresponding directed edge is not in L. The sums are over all $v_j \in G$. Notice that the equation still holds if the sums are over $v_j \in L$. In another word, the flow conservation condition is satisfied by L with respect to G as well as with respect to L.
2. $G \setminus L$ satisfies the flow conservation condition (1) at every node $v_i \in G$.
   Let w_{ij} be the edge weight for G, w'_{ij} be the edge weight for $G \setminus L$, $w_{ij}^L = w$ be the edge weight for L. From the above, for any vertex $v_i \in G$,

$$\sum_j w'_{ij} = \begin{cases} \sum_j w_{ij} - w = \sum_j w_{ji} - w = \sum_j w'_{ji} & \text{if the vertex } v_i = v_{i_m} \in L \\ \sum_j w_{ij} = \sum_j w_{ji} = \sum_j w'_{ji} & \text{if the vertex } v_i \notin L \end{cases}$$

Consequently from the above facts (1) and (2), $G_r = G \setminus \bigcup_{i=1}^r L$ satisfies the flow conservation condition.

Let $w_{ij}^{G_r}$ denote the edge weight for the reminder graph G_r. By definition, G_r contains no simple cycles, particularly no one-cycles, or self-loops, of the form $v_i \leftrightarrow v_i$. That is, $w_{ii}^{G_r} = 0, \forall i$. Also G_r contains no two-cycles $v_i \to v_j \to v_i$, which implies $w_{ij}^{G_r} = 0$ if $w_{ji}^{G_r} \neq 0$.

We claim that $w_{ij}^{G_r} \equiv 0$. Assume instead that there is at least one $w_{ij}^{G_r} > 0$ on G_r. Let

$$w_o = w_{r_2, r_1}^{G_r} = \min_{i \neq j} \left\{ w_{ij}^{G_r} : w_{ij}^{G_r} > 0 \right\}$$

be the smallest non-zero weight on G_r, where $w_o = w_{r_2,r_1}$ means that the directed edge from vertex v_{r_1} to vertex $v_{r_2} \neq v_{r_1}$ is of weight w_o. Since G_r satisfies the flow conservation condition at vertex v_{r_2},

$$\sum_j w_{j,r_2}^{G_r} = \sum_j w_{r_2,j}^{G_r} \geq w_{r_2,r_1}^{G_r} = w_o > 0.$$

There must be a vertex v_{r_3} such that $w_{r_3,r_2}^{G_r} \geq w_o > 0$, i.e., there is an edge with weight $w_{r_3,r_2}^{G_r}$ from vertex v_{r_2} to vertex v_{r_3}, and $v_{r_1}, v_{r_2}, v_{r_3}$ are distinct because G_r contains no more simple cycles. At vertex v_{r_3}, G_r satisfies the flow conservation condition,

$$\sum_j w_{j,r_3}^{G_r} = \sum_j w_{r_3,j}^{G_r} \geq w_{r_3,r_2}^{G_r} \geq w_o > 0$$

and so on. Because there are no simple cycles of any length in G_r, this procedure would construct a path $v_{r_1} \to v_{r_2} \to \cdots \to v_{r_m}$ ended at a vertex v_{r_m} for some $m > 1$ with $< n$ edges in the path. This implies

$$\sum_j w_{j,r_m}^{G_r} = 0 \neq \sum_j w_{r_m,j}^{G_r} \geq w_{r_m,r_{m-1}}^{G_r} \geq w_o > 0$$

The conservation condition could not be satisfied at the end vertex v_{r_m}. This is a contradiction. Therefore there must be $w_{ij}^{G_r} \equiv 0$. Consequently $G_r = \emptyset$. This completes the proof of Corollary 2.1.

Corollary 2.2 *For the remainder graph G_r, all eigenvalues of its weight matrix $\left(w_{ij}^{G_r}\right)_{n\times n}$ are zero.*

Proof Consider the nontrivial case $G_r \neq \emptyset$. An eigenvalue λ of $\left(w_{ij}^{G_r}\right)_{n\times n}$ makes the corresponding characteristic function zero, i.e., $det\left\{\lambda I_n - \left(w_{ij}^{G_r}\right)_{n\times n}\right\} = 0$, where I_n is the identity matrix of order n. Consider an expansion of the characteristic function

$$det\left\{\lambda I_n - \left(w_{ij}^{G_r}\right)_{n\times n}\right\} = \lambda^n + c_1\lambda^{n-1} + c_2\lambda^{n-2} + \cdots + c_n$$

with

$$c_1 = w_{11}^{G_r} + w_{22}^{G_r} + \cdots + w_{nn}^{G_r}$$

$$c_2 = -\sum_{\binom{i_1\ i_2}{j_1\ j_2}\in S_n} sign\begin{pmatrix} i_1 & i_2 \\ j_1 & j_2 \end{pmatrix} w_{i_1,j_1}^{G_r} w_{i_2,j_2}^{G_r}$$

$$c_3 = -\sum_{\left(\begin{smallmatrix} i_1 & i_2 & i_3 \\ j_1 & j_2 & j_3 \end{smallmatrix}\right) \in S_n} sign \begin{pmatrix} i_1 & i_2 & i_3 \\ j_1 & j_2 & j_3 \end{pmatrix} w^{G_r}_{i_1,j_1} w^{G_r}_{i_2,j_2} w^{G_r}_{i_3,j_3}$$

$$\vdots$$

$$c_n = -\sum_{\left(\begin{smallmatrix} i_1 & i_2 & \cdots & i_n \\ j_1 & j_2 & \cdots & j_n \end{smallmatrix}\right) \in S_n} sign \begin{pmatrix} i_1 & i_2 & \cdots & i_n \\ j_1 & j_2 & \cdots & j_n \end{pmatrix} w^{G_r}_{i_1,j_1} w^{G_r}_{i_2,j_2} \cdots w^{G_r}_{i_n,j_n}$$

where S_n is the symmetric group of order n, its element

$$\begin{pmatrix} i_1 & i_2 & \cdots & i_n \\ j_1 & j_2 & \cdots & j_n \end{pmatrix} \in S_n$$

is a permutation of ordered arrangement of integers $\{1\ 2\ 3\ \cdots\ n\}$ from $\{i_1\ i_2\ i_3\ \cdots\ i_n\}$ to $\{j_1\ j_2\ j_3\ \cdots\ j_n\}$. The notation

$$\begin{pmatrix} i_1 & i_2 & \cdots & i_k \\ j_1 & j_2 & \cdots & j_k \end{pmatrix} \in S_n, \quad 1 < k < n$$

denotes a subset permutation of k of the n integers $\{1\,2\,3\ \cdots\ n\}$ from $\{i_1\ i_2\ \cdots\ i_k\}$ to $\{j_1\ j_2\ \cdots\ j_k\}$, while the other $n-k$ integers $\{1\,2\,3\ \cdots\ n\} \setminus \{i_1\ i_2\ \cdots\ i_k\}$ remain unchanged. A switch of exactly two integers

$$\begin{pmatrix} \cdots & i & \cdots & i' & \cdots \\ \cdots & i' & \cdots & i & \cdots \end{pmatrix}$$

is considered as one move in S_n. The sign of a permutation is -1 if the number of moves needed to complete the permutation is odd, $+1$ otherwise.

Since there are no simple cycles in G_r, there are no one-cycles, or self-loops $v_i \leftrightarrow v_i$. Therefore

$$w^{G_r}_{ii} \equiv 0 \quad \forall i = 1, \ldots, n$$

which implies

$$c_1 = 0.$$

Consequently, if

$$\exists\, i_m = j_m,\ 1 \le m \le k \quad \text{in} \quad \begin{pmatrix} i_1 & i_2 & \cdots & i_k \\ j_1 & j_2 & \cdots & j_k \end{pmatrix}, \quad 1 < k \le n$$

then the corresponding term

$$w^{G_r}_{i_1,j_1} w^{G_r}_{i_2,j_2} \cdots w^{G_r}_{i_k,j_k} = 0.$$

For $k = 2$, the above implies that the terms in c_2 must have the form

$$c_2 = - \sum_{\binom{i_1\ i_2}{i_2\ i_1} \in S_n} sign \begin{pmatrix} i_1 & i_2 \\ i_2 & i_1 \end{pmatrix} w^{G_r}_{i_1,i_2} w^{G_r}_{i_2,i_1}$$

since the terms $w^{G_r}_{i_1,i_2} w^{G_r}_{i_2,i_1}$ corresponding to permutations

$$\begin{pmatrix} i_1 & i_2 \\ i_1 & i_2 \end{pmatrix}$$

are zero. Furthermore, there are no two-cycles $v_{i_1} \to v_{i_2} \to v_{i_1}$ in G_r, thus

$$w^{G_r}_{i_1,i_2} w^{G_r}_{i_2,i_1} \equiv 0 \quad \forall i_1, i_2 = 1, \ldots, n$$

which implies

$$c_2 = 0.$$

And consequently, if

$$\exists\, i_m = j_{m'},\ i_{m'} = j_m, \quad 1 \le m,\ m' \le k$$

in

$$\begin{pmatrix} i_1 & \cdots & i_m & \cdots & i_{m'} & \cdots & i_k \\ j_1 & \cdots & j_m & \cdots & j_{m'} & \cdots & j_k \end{pmatrix}, \quad 1 < k \le n$$

then the corresponding term

$$w^{G_r}_{i_1,j_1} \cdots w^{G_r}_{i_m,j_m} \cdots w^{G_r}_{i_{m'},j_{m'}} \cdots w^{G_r}_{i_k,j_k} = w^{G_r}_{i_1,j_1} \cdots w^{G_r}_{i_m,j_m} \cdots w^{G_r}_{j_m,i_m} \cdots w^{G_r}_{i_k,j_k} = 0.$$

The same argument on k-cycles gives

$$w^{G_r}_{i_1,i_k} w^{G_r}_{i_k,i_{k-1}} \cdots w^{G_r}_{i_2,i_1} \equiv 0$$

for $k = 3, 4, \ldots, n$, which implies

$$c_k = 0, \quad \forall k = 3, 4, \ldots, n.$$

Therefore the characteristic function

$$\begin{aligned} &\lambda^n + c_1\lambda^{n-1} + c_2\lambda^{n-2} + \cdots + c_n = \lambda^n = 0 \\ \Longrightarrow \quad &\lambda = 0. \end{aligned}$$

This completes the proof of Corollary 2.2.

3 Illustrative examples

In this section, three simple, hypothetical examples are used to illustrate the properties of the algorithm and the decomposition. The numbers used are for illustration convenience, they do not correspond to realistic population dynamics situations.

Example 3.1 – A simple decomposition with non-empty remainder graph
Consider the weighted digraph in Fig. 1.

The corresponding matrix of directed weights is

$$\begin{array}{c} \\ \text{from} \\ [1]\ \ [2]\ \ [3] \end{array}$$

$$\begin{array}{r} \\ [1] \\ \text{to}\ [2] \\ [3] \end{array} \begin{bmatrix} 13 & 12 & 15 \\ 26 & 18 & 0 \\ 0 & 15 & 2 \end{bmatrix} = G$$

We begin by searching for simple cycles starting from the node **1**.

- Start with node **1**, find a positive cycle in G containing node **1**. We get $L_1 = \{\mathbf{1} \to \mathbf{2} \to \mathbf{1}\}$.
- Take the smallest edge weight $w_{12}^{G} = 12$ in the selected cycle as the weight for all edges of L_1.
- Remove the cycle L_1 from the graph G. The remaining graph G_1 is shown in Fig. 2.
- Start with node **1**, find another cycle in G_1 containing node **1**. We get $L_2 = \{\mathbf{1} \to \mathbf{2} \to \mathbf{3} \to \mathbf{1}\}$.
- Take the smallest edge weight $w_{21}^{G_1} = 14$ in the cycle L_2 as the weight for all edges in the cycle.
- Remove the cycle L_2 from the graph G_1. The remaining graph G_2 is shown in Fig. 3.
- The remaining graph G_2 contains three self-loops, or one-cycles. Remove the self-loops L_3, L_4 and L_5 from G_2. The remaining graph G_5 is shown in Fig. 4.

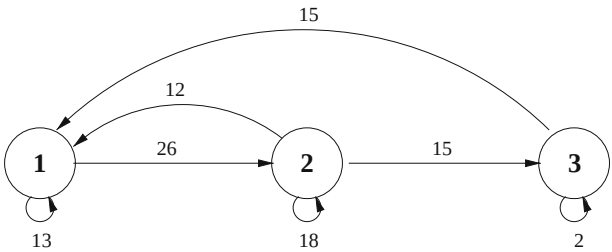

Fig. 1 Example 3.1, original graph G

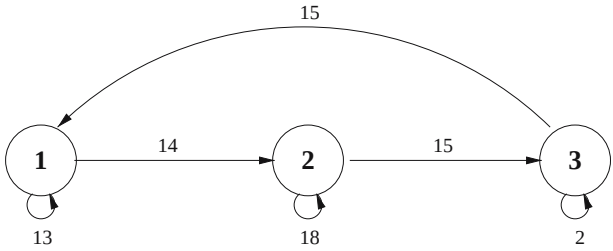

Fig. 2 Example 3.1, subgraph G_1

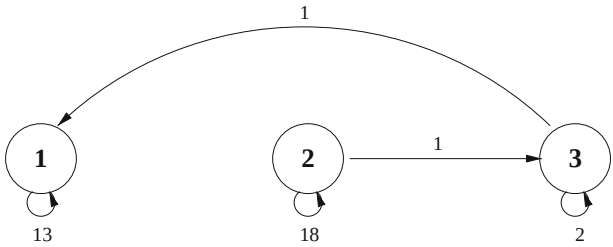

Fig. 3 Example 3.1, subgraph G_2

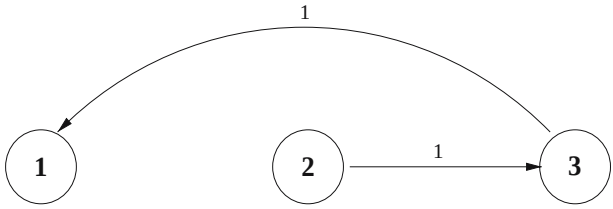

Fig. 4 Example 3.1, remainder graph G_5

– G_5 contains no cycles. The original graph is decomposed into five cycles and a remainder graph G_5.

$$G = G_5 \bigcup \left(\bigcup_{i=1}^{5} L_i \right)$$

– The decomposition steps can be written in terms of a decomposition of the corresponding matrix. To remove L_1 from G,

$$G = \begin{bmatrix} 13 & 12 & 15 \\ 26 & 18 & 0 \\ 0 & 15 & 2 \end{bmatrix} = \begin{bmatrix} 0 & 12 & 0 \\ 12 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 13 & 0 & 15 \\ 14 & 18 & 0 \\ 0 & 15 & 2 \end{bmatrix} = L_1 + G_1$$

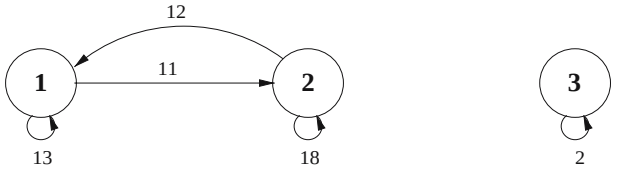

Fig. 5 Example 3.2, subgraph G_1

To remove L_2 from G_1,

$$\begin{bmatrix} 13 & 12 & 15 \\ 26 & 18 & 0 \\ 0 & 15 & 2 \end{bmatrix} = \begin{bmatrix} 0 & 12 & 0 \\ 12 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 14 \\ 14 & 0 & 0 \\ 0 & 14 & 0 \end{bmatrix} + \begin{bmatrix} 13 & 0 & 1 \\ 0 & 18 & 0 \\ 0 & 1 & 2 \end{bmatrix}$$

To complete the decomposition,

$$G = \begin{bmatrix} 13 & 12 & 15 \\ 26 & 18 & 0 \\ 0 & 15 & 2 \end{bmatrix} = \begin{bmatrix} 0 & 12 & 0 \\ 12 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 14 \\ 14 & 0 & 0 \\ 0 & 14 & 0 \end{bmatrix} + \begin{bmatrix} 13 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 18 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 2 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

$$G = \{\mathbf{1} \to \mathbf{2} \to \mathbf{1}\} \bigcup \{\mathbf{1} \to \mathbf{2} \to \mathbf{3} \to \mathbf{1}\} \bigcup \{\text{self loops}\} \bigcup G_5$$

Notice that, the corresponding characteristic function of the remainder graph $G_r = G_5$ is

$$det \begin{bmatrix} \lambda & 0 & -1 \\ 0 & \lambda & 0 \\ 0 & -1 & \lambda \end{bmatrix} = \lambda^3$$

All eigenvalues of G_r are zero.

Example 3.2 – Non-uniqueness of the decomposition

Consider the same weighted digraph G in Fig. 1 of Example 3.1. This time we extract the three-cycle $\{\mathbf{1} \to \mathbf{2} \to \mathbf{3} \to \mathbf{1}\}$ first. The first remaining graph G_1 is Fig. 5.

Notice that the edge weight for the cycle $\{\mathbf{1} \to \mathbf{2} \to \mathbf{3} \to \mathbf{1}\}$ is 15 instead of 14 as in Example 3.1. Next we extract the two-cycle $\{\mathbf{1} \to \mathbf{2} \to \mathbf{1}\}$. The edge weight for the cycle is 11 instead of 12 as in Example 3.1. The remaining subgraph G_2 is Fig. 6.

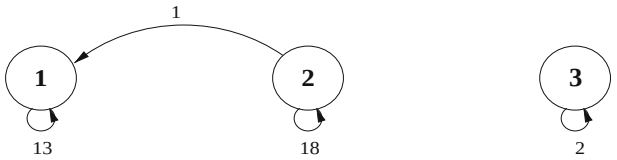

Fig. 6 Example 3.2, subgraph G_2

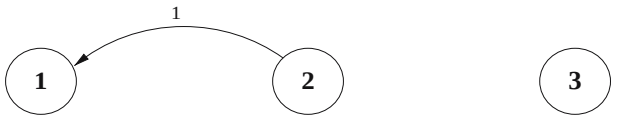

Fig. 7 Example 3.2, remainder graph G_5

Removing the self loops will end up with an acyclic graph G_5 (Fig. 7).
The corresponding matrix decomposition is

$$\begin{bmatrix} 13 & 12 & 15 \\ 26 & 18 & 0 \\ 0 & 15 & 2 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 15 \\ 15 & 0 & 0 \\ 0 & 15 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 11 & 0 \\ 11 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 13 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$
$$+ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 18 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 2 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

This alternative decomposition can be written as

$$G = \{\mathbf{1} \to \mathbf{2} \to \mathbf{3} \to \mathbf{1}\} \bigcup \{\mathbf{1} \to \mathbf{2} \to \mathbf{1}\} \bigcup \{\text{self loops}\} \bigcup G_5$$

The decomposition results in the same set of cycles as in the previous example, but the edge weights are different. Although the remainder graph here is different from the remainder graph in the previous example, its characteristic function is

$$det \begin{bmatrix} \lambda & -1 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \lambda \end{bmatrix} = \lambda^3$$

Again all eigenvalues of G_r are zero.

Example 3.3 – A complete decomposition into cycles
Consider the weighted digraph in Fig. 8.
This is the graph in Example 3.1 with weight $w_{12}^G = 11$ instead of 12. However this modified graph satisfies the flow conservation condition

$$\sum_{j=1}^{3} w_{ij} = \sum_{j=1}^{3} w_{ji}, \quad \forall i = 1,2,3$$

This can be seen more clearly from the row and column sums of the corresponding matrix:

$$\begin{array}{cc} & \text{row sum} \\ \begin{bmatrix} 13 & 11 & 15 \\ 26 & 18 & 0 \\ 0 & 15 & 2 \end{bmatrix} & \begin{matrix} 39 \\ 44 \\ 17 \end{matrix} \\ \text{column sum} \quad 39\ 44\ 17 & \end{array}$$

The decomposition will be complete, i.e. $G_r = \emptyset$, the empty graph. The matrix decomposition can be written as

$$G = \begin{bmatrix} 13 & 12 & 15 \\ 26 & 18 & 0 \\ 0 & 15 & 2 \end{bmatrix} = \begin{bmatrix} 0 & 11 & 0 \\ 11 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 15 \\ 15 & 0 & 0 \\ 0 & 15 & 0 \end{bmatrix} + \begin{bmatrix} 13 & 0 & 0 \\ 0 & 18 & 0 \\ 0 & 0 & 2 \end{bmatrix}$$

The graph is decomposed into five cycles:

$$G = \{\mathbf{1} \to \mathbf{2} \to \mathbf{1}\} \bigcup \{\mathbf{1} \to \mathbf{2} \to \mathbf{3} \to \mathbf{1}\} \bigcup \{\text{self loops}\}$$

In this example, the decompositions are the same whether one starts from the long cycle $\{\mathbf{1} \to \mathbf{2} \to \mathbf{3} \to \mathbf{1}\}$ or from the short cycle $\{\mathbf{1} \to \mathbf{2} \to \mathbf{1}\}$.

4 Applications in life cycle analysis

In this section, we apply the proposed method to life cycle analysis, more specifically loop analysis, in population dynamics studies.

4.1 Matrix population models

In the analysis of population structures and population dynamics, matrix population models have been used on a wide range of species. A typical population projection matrix is of the form

$$\mathbf{N}(t+1) = \begin{bmatrix} N_1(t+1) \\ N_2(t+1) \\ \vdots \\ N_n(t+1) \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} N_1(t) \\ N_2(t) \\ \vdots \\ N_n(t) \end{bmatrix} = A\,\mathbf{N}(t)$$

where n is the number of life cycle stages and $N_i(t)$ is the number of individuals of the population in stage i at time t. The stages are phases in the life cycle, such as age and size, identified as potentially having high impact on the population growth rate. The projection matrix element a_{ij} is the contribution of individuals in stage j at time t to the population in stage i at time $t+1$, i.e., the proportion of stage j individuals that survive and grow into stage i plus the rate of successful birth by stage j individuals into stage i from time t to time $t+1$. Based on Perron–Frobenius theory and its variations [3] on non-negative, irreducible and primitive matrices, the matrix A has a dominating eigenvalue (i.e., one of largest norm) $\lambda > 0$. Biologically, λ is the long term growth rate of the population.

4.2 Sensitivity and elasticity matrices

The matrix elements a_{ij} are composed of life cycle parameters such as survival rate, growth rate and birth rate of individuals in a stage. The impact of life cycle parameters on the population growth rate can be analyzed via the sensitivity matrix $S = \left(s_{ij}\right)_{n\times n}$, $s_{ij} = \frac{\partial\lambda}{\partial a_{ij}}$ and the elasticity matrix $E = \left(e_{ij}\right)_{n\times n}, e_{ij} = \frac{\partial \ln\lambda}{\partial \ln a_{ij}}$ evaluated at the long term population rate. The analysis is particularly useful in demographic analysis [10–12]. The relationship between S and E is $e_{ij} = \frac{\lambda}{a_{ij}} s_{ij}$. Elasticities are also called proportional sensitivities. The total elasticity (the sum of the elements in E) is one, or 100%. In addition, elasticity is *conserved* at each stage:

$$\sum_{j=1}^{n} e_{ij} = \sum_{j=1}^{n} e_{ji} \quad \forall i = 1, \ldots, n,$$

i.e., the elasticity matrix satisfies the flow conservation condition. In life cycle analysis, elasticity can be viewed as a conservative quantity that "flows" through the life cycle graph. When the population life cycle graph decomposes into different cycles that represent life paths followed by different individual organisms (as in loop analysis described in the section below), the total elasticity of each cycle represents the proportional sensitivity of the population growth rate λ to the particular life path. In other words, elasticities can be used to describe the relative contribution of alternative life paths to variations in total population growth rate λ.

4.3 Loop analysis

Loop analysis is a type of sensitivity analysis for demographic models. For example, Fig. 8 in Example 3.3 can be viewed as a graphical representation of a demographic model – a life cycle graph. The nodes {**1, 2, 3**} may represent three life stages, e.g., {*baby, youth, adult*}, or {*small, medium, large*}. The edges represent directed transitions between stages. A population projection

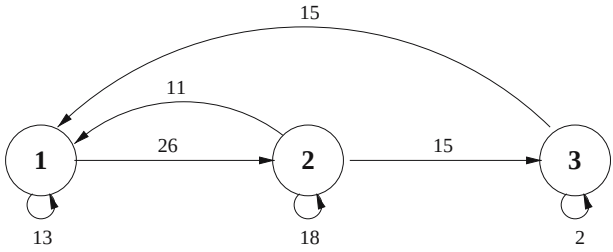

Fig. 8 Example 3.3, original graph G

matrix describes the transition rate between stages. Loop analysis, or life cycle analysis, focuses on the life cycles, their fates, and their contributions to total population growth rate. Life cycles (simple cycles in graph theory), such as $\{\mathbf{1} \to \mathbf{2} \to \mathbf{3} \to \mathbf{1}\}$ and $\{\mathbf{1} \to \mathbf{2} \to \mathbf{1}\}$, are called *loops*. Each life cycle is a path followed by some individuals in the population. In loop analysis, the weights of the directed edges in the life cycle graph are the elasticity values. To conduct loop analysis, the life cycle graph is decomposed into a set of loops. Loop elasticity is defined as the sum of the weights of the edges in the loop. Since elasticity matrix satisfies the flow conservation condition, biologically, loop elasticity of a life cycle can be interpreted as the proportional sensitivity of the population growth rate to the life cycle. The flow conservation condition also implies that loop elasticities are additive. The total elasticity of all loops sums to 100%. Loop elasticities of different life cycles can be used to compare the relative contributions of the various life cycles to changes in the population growth rate.

A graph theoretic spanning-tree method can be used to decompose the life cycle graph into a set of loops. The method was formulated and illustrated by Wardle [13] with detail and clarity. Wardle also pointed out problems in using the spanning tree method. Primarily, it may be hard or impossible to avoid loops containing contradictory directions when decomposing complex life cycle graphs. The following examples present such situations and illustrate our proposed decomposition method.

4.4 Examples

Example 4.1 – A classic example (teasel *Dipsacus sylvestris*)

This example was used in Wardle [13] to illustrate loops with contradictory directions. Wardle also proposed methods to deal with such loops. The life cycle graph (Fig. 9) is reproduced based on the version in [13], originally from Werner [14], Werner and Caswell [15], Caswell and Werner [4] and later reanalyzed by Caswell [3].

Dipsacus sylvestris is a monocarpic (bearing fruit but once, and dying after fructification), perennial plant. In the life cycle graph, the stages 1–6 represent: first year dormant seeds, second year dormant seeds, small rosettes, medium rosettes, large rosettes, and flowering individuals. The weights of the directed

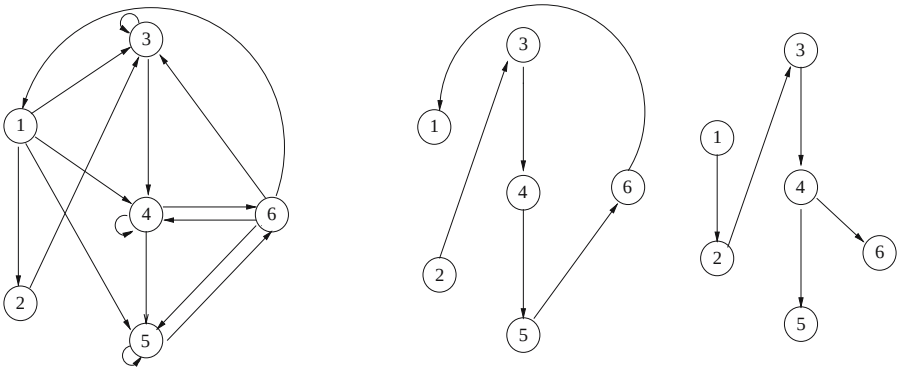

Fig. 9 Example 4.1, the life cycle graph G and two spanning trees

edges are the elasticities (not shown in the figure). The original elasticity matrix ($\times 100$) is

stage	(1)	(2)	(3)	(4)	(5)	(6)	row sum
(1)	0	0	0	0	0	6.594	6.594
(2)	0.025	0	0	0	0	0	0.025
(3)	0.079	0.025	0.015	0	0	0.151	0.270
(4)	0.750	0	0.256	2.773	0	23.270	27.049
(5)	5.740	0	0	19.120	2.272	4.454	31.586
(6)	0	0	0	5.157	29.310	0	34.467
column sum	6.594	0.025	0.271	27.050	31.582	34.469	99.991

(2)

The life cycle graph is complicated. There are no known spanning trees that can produce a set of loops containing no contradictory directions. Wardle [13] selected a set that contains one loop with contradictory directions, using the first spanning tree $\mathbf{2} \Rightarrow \mathbf{3} \Rightarrow \mathbf{4} \Rightarrow \mathbf{5} \Rightarrow \mathbf{6} \Rightarrow \mathbf{1}$ in Fig. 9.

A spanning tree of a graph is an acyclic subgraph connecting all n nodes in the original graph. Notice that a spanning tree always consists of $n - 1$ edges. Two possible spanning trees are depicted in Fig. 9. The *co-tree* consists of the $|E(G)| - (n - 1)$ edges not used by the spanning tree, where $|E(G)|$ is the total number of edges in the original graph G.

A loop is formed by adding an edge from the co-tree to the spanning tree. For example, adding the edge $\mathbf{1} \rightarrow \mathbf{3}$ to the first spanning tree produces the loop $\{\mathbf{1} \rightarrow \mathbf{3} \Rightarrow \mathbf{4} \Rightarrow \mathbf{5} \Rightarrow \mathbf{6} \Rightarrow \mathbf{1}\}$, where double arrows represent edges in the spanning tree and the single arrow is the edge from the co-tree. The weight of the edge from the co-tree is the *characteristic elasticity* of the loop. The loop elasticity is the characteristic elasticity multiplied by the length of the loop. Therefore, the loop $\{\mathbf{1} \rightarrow \mathbf{3} \Rightarrow \mathbf{4} \Rightarrow \mathbf{5} \Rightarrow \mathbf{6} \Rightarrow \mathbf{1}\}$ has characteristic elasticity 0.079 and loop elasticity $0.079 \times 5 = 0.395$.

Each spanning tree uniquely determines a set of independent loops because each loop contains a unique edge from the co-tree. The problematic loop for the first tree occurs when the edge $\mathbf{4} \rightarrow \mathbf{6}$ is added to the tree. The loop formed is

$$\{\mathbf{4} \Rightarrow \mathbf{5} \Rightarrow \mathbf{6} \leftarrow \mathbf{4}\} \tag{3}$$

The loop contains contradictory directions. There are no biological interpretations for such loops. No individuals in the population follow the path of the loop as a life cycle. Other spanning trees for this example do not avoid producing loops with contradictory directions. Wardle proposed three methods to handle the problem after the set of loops is produced.

To use our proposed algorithm to decompose the life cycle graph (Fig. 9), we start with the elasticity matrix. Theoretically, the elasticity matrix should satisfy the flow conservation condition. However, in the given elasticity matrix (2), the flow conservation condition is only approximately satisfied because of rounding errors. We modify the elements (3,6), (4,6) and (5,6) slightly to obtain a matrix that satisfies the flow conservation condition:

stage	(1)	(2)	(3)	(4)	(5)	(6)	row sum
(1)	0	0	0	0	0	6.594	6.594
(2)	0.025	0	0	0	0	0	0.025
(3)	0.079	0.025	0.015	0	0	0.152	0.271
(4)	0.750	0	0.256	2.773	0	23.271	27.050
(5)	5.740	0	0	19.120	2.272	4.450	31.582
(6)	0	0	0	5.157	29.310	0	34.467
column sum	6.594	0.025	0.271	27.050	31.582	34.467	99.989

(4)

The difference between different modifications is usually negligible as long as sufficient significant digits are left untouched. The elasticities in (4) are assigned as the weights of the directed edges in the life cycle graph of Fig. 9. Using the proposed algorithm, we decompose the graph into a set of loops with no contradictory directions (Table 1).

The step by step decomposition of the elasticity matrix (5) is in the Appendix. The total number of loops is $11 = \text{edges} - \text{nodes} + 1 = 11 - 6 + 1$, which equals the nullity of the life cycle graph G of Fig. 9. Table 1 exhibits one set of loops listed in the order searched. The loops can be further grouped [13] to represent plants that are: (a) with 2 years in seed bank; (b) with 1 year in seed bank; (c) quadrennials; (d) triennials; (e) biennials; and (f) with delays at rosette stages. Different search orders could produce different loops. Here the loops with starting stage **1** were given higher priorities. Search order does not affect the loop elasticities of the self-loops L_9, L_{10}, L_{11}. Notice that the loop L_7 cannot be produced by the first spanning tree in Fig. 9. In one of the methods (Method C) proposed by Wardle [13] to handle the loop with contradictory directions, loop (3) was changed into L_7, the loop elasticity of another loop (corresponding to L_6) was adjusted accordingly to keep total elasticity unchanged. Other than

Table 1 Example 4.1 loops, the removed edges of each loop, characteristic elasticities and loop elasticities

Graph	Simple cycle (loop)	Removed edge of min wgt	Minimum wgt char. elasticity	Loop wgt loop elasticity	Sum
(a)	L_1={1→ 2→ 3→ 4→ 5→ 6→1}	(2,1)	0.025	0.150	0.150
(b)	L_2={1 → 3 → 4 → 5 → 6 → 1}	(3,1)	0.079	0.395	20.615
	L_3 = {1 → 4 → 5 → 6 → 1}	(4,1)	0.750	3.000	
	L_4 = {1 → 5 → 6 → 1}	(5,1)	5.740	17.220	
(c)	L_5 = {3 → 4 → 5 → 6 → 3}	(4,3)	0.152	0.608	0.608
(d)	L_6 = {4 → 5 → 6 → 4}	(5,4)	18.114	54.342	54.342
(e)	L_7 = {4 → 6 → 4}	(6,4)	5.157	10.314	19.214
	L_8 = {5 → 6 → 5}	(6,5), (5,6)	4.450	8.900	
(f)	L_9 = {3 ↔ 3}	(3,3)	0.015	0.015	5.060
	L_{10} = {4 ↔ 4}	(4,4)	2.773	2.773	
	L_{11} = {5 ↔ 5}	(5,5)	2.272	2.272	
	Total			99.989	99.989

small differences due to our adjustment of the rounding errors in the elasticity matrix, the set of loops listed in the above table is consistent with the set of modified loops obtained in [13] by Method C.

Example 4.2 – A new challenge

The life cycle graph in Fig. 10 based on a model for a population of kelp (*Alaria Nana*, [8]) motivated the development of the proposed decomposition method.

Stages **1, 3, 5** represent slow growers with sizes small, medium and large, and **2, 4, 6** represent fast growers with sizes small, medium and large. Dotted lines indicate reproductions. The model is inspired by the research of Pfister and Stevens [8] and is developed further in [9]. In this example, we focus on the mathematical issues in the decomposition, using an elasticity matrix based on one set of estimated parameters. For readability, the following matrix consists of elements of the elasticity matrix multiplied by 1,000. Routine modification has been applied to the matrix to preserve the flow conservation condition.

stage	(1)	(2)	(3)	(4)	(5)	(6)	row sum
(1)	79	21	2	1	12	10	125
(2)	34	74	2	2	16	15	143
(3)	9	14	22	20	0	0	65
(4)	3	34	14	25	0	0	76
(5)	0	0	15	12	190	99	316
(6)	0	0	10	16	98	152	276
column sum	125	143	65	76	316	276	1001

(5)

For the life cycle graph in Fig. 10, the number of independent loops equals the nullity $23 = 28 - 6 + 1$. There are considerable difficulties in using the spanning tree method to produce a set of loops. The following trees

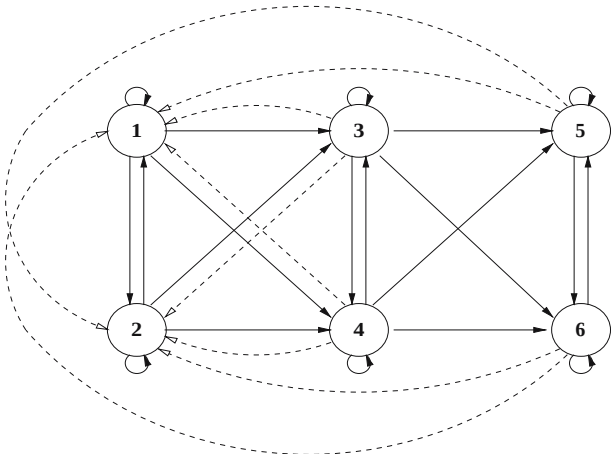

Fig. 10 Example 4.2 life cycle graph

$$\begin{array}{ccccc} \mathbf{1} & \Rightarrow & \mathbf{3} & \Rightarrow & \mathbf{5} \\ \Downarrow & & & & \\ \mathbf{2} & \Rightarrow & \mathbf{4} & \Rightarrow & \mathbf{6} \end{array} \tag{6}$$

and

$$\begin{array}{ccccc} \mathbf{1} & \Rightarrow & \mathbf{3} & \Rightarrow & \mathbf{5} \\ \Uparrow & & & & \\ \mathbf{2} & \Rightarrow & \mathbf{4} & \Rightarrow & \mathbf{6} \end{array} \tag{7}$$

produce several pairs of loops that are biologically interesting, such as $\{\mathbf{1} \Rightarrow \mathbf{3} \Rightarrow \mathbf{5} \rightarrow \mathbf{1}\}$ and $\{\mathbf{2} \Rightarrow \mathbf{4} \Rightarrow \mathbf{6} \rightarrow \mathbf{2}\}$, representing life cycles of individuals with a definite growth status. However each set spanned by the trees (6) or (7) contains ten loops with contradictory directions. The spanning tree

$$\mathbf{2} \Rightarrow \mathbf{3} \Rightarrow \mathbf{4} \Rightarrow \mathbf{5} \Rightarrow \mathbf{6} \Rightarrow \mathbf{1}$$

produces fewer loops (four) with contradictory directions. However many loops of biological interest do not show up in this set, and the number of loops with contradictory directions are still too numerous to be handled by the modification method in [13]. Other spanning trees considered are inferior in terms of the production of many loops with contradictory directions and the inability to obtain pairs of loops of biological interest.

Combining stages is another way to handle loops with contradictory directions. We also considered combining stages **1** and **2**, or stages **5** and **6**. Certain spanning trees can produce a few pairs of loops of moderate interest. However, loops with contradictory directions spring up persistently.

Table 2 Example 4.2 loops, the removed edges of each loop, characteristic elasticities and loop elasticities

Simple cycle (loop)	Removed edge of min weight	Minimum weight characteristic elasticity	Loop weight loop elasticity
$L_1 = \{1 \to 3 \to 1\}$	(1,3)	2	4
$L_2 = \{2 \to 4 \to 2\}$	(2,4)	2	4
$L_3 = \{1 \to 3 \to 5 \to 1\}$	(3,1)	7	21
$L_4 = \{2 \to 4 \to 6 \to 2\}$	(2,6)	15	45
$L_5 = \{1 \to 4 \to 6 \to 1\}$	(6,4)	1	3
$L_6 = \{1 \to 4 \to 5 \to 1\}$	(1,4)	2	6
$L_7 = \{1 \to 2 \to 3 \to 5 \to 1\}$	(1,5)	3	12
$L_8 = \{1 \to 2 \to 3 \to 6 \to 1\}$	(1,6)	9	36
$L_9 = \{1 \to 2 \to 4 \to 1\}$	(1,4)	1	3
$L_{10} = \{2 \to 3 \to 5 \to 2\}$	(3,2)	2	6
$L_{11} = \{2 \to 4 \to 5 \to 2\}$	(4,5)	10	30
$L_{12} = \{2 \to 4 \to 3 \to 5 \to 2\}$	(5,3)	3	12
$L_{13} = \{2 \to 4 \to 3 \to 6 \to 5 \to 2\}$	(2,5)	1	5
$L_{14} = \{2 \to 4 \to 3 \to 2\}$	(4,2)	2	6
$L_{15} = \{1 \rightleftharpoons 2\}$	(1,2)	21	42
$L_{16} = \{3 \rightleftharpoons 4\}$	(4,3)	14	28
$L_{17} = \{5 \rightleftharpoons 6\}$	(6,5)	98	196
$L_{18} = \{1 \leftrightarrow 1\}$	(1,1)	79	79
$L_{19} = \{2 \leftrightarrow 2\}$	(2,2)	74	74
$L_{20} = \{3 \leftrightarrow 3\}$	(3,3)	22	22
$L_{21} = \{4 \leftrightarrow 4\}$	(4,4)	25	25
$L_{22} = \{5 \leftrightarrow 5\}$	(5,5)	190	190
$L_{23} = \{6 \leftrightarrow 6\}$	(6,6)	152	152
Total			1001

Using the proposed method, we obtain a set of loops with no contradictory directions. Also, pairs of loops (e.g., L_1 and L_2, L_3 and L_4) that are biologically interesting appear in the same set (Table 2).

Since the data and parameter estimates used here are intermediate results from a work in progress [9], we restrain our comments to mathematically relevant matters. The loops are independent. The total number of loops in the set is 23, which equals the nullity of the graph. The decomposition gives priorities to loops of interest ($L_1 - L_4$) and loops started at stages **1** and **2**. Corollary 2.1 asserts the complete decomposition of the matrix, regardless of loop selections. This set of loops cannot be produced from any spanning tree. The decomposition of the elasticity matrix (5) is similar to that of Example 4.1. Details of the decomposition are in the Appendix.

5 Conclusion

5.1 Biological implications

In the study of population dynamics, loop analysis has been used for comparing the relative importance of different life paths to the population growth rate. An essential step of loop analysis is to decompose the life cycle graph of the

population into a set of life cycles followed by individuals in the population. A graph theoretic spanning tree method has been used to provide a systematic approach to the decomposition. The method provides a set of loops with elasticities summing to 1. However there are difficulties in realizing a suitable decomposition for complex life histories using the spanning tree method. One of the problems is the occurrence of life cycles that contain contradictory directions, caused by the existence of two or more pairs of life stages with reversions (e.g., the life stage pairs $4 \rightleftharpoons 5$ and $5 \rightleftharpoons 6$ in Example 4.1, and the pairs $2 \rightleftharpoons 3$, $2 \rightleftharpoons 4$ and $3 \rightleftharpoons 4$ in Example 4.2). Cycles with contradictory directions are unavoidable for some complex life cycle graphs: there may not exist any tree that spans a set of cycles containing no contradictory directions. There is no biological interpretation for such cycles, since a cycle with contradictory directions generally cannot represent the life path followed by individual organisms. Ad hoc method to modify or eliminate cycles with contradictory directions have not been satisfactory. The proposed algorithm guarantees a complete decomposition of a population life cycle graph into a set of life cycles that do not contain contradictory directions. As in the spanning tree method, the decomposition is generally not unique, and thus important, meaningful life cycles should be given higher priority for selection by the algorithm.

5.2 The algorithm and its properties

We propose an algorithmic searching procedure for decomposing a directed, weighted graph. This proposed approach can be viewed as the spanning tree method reversed.

In the spanning tree method, arcs are successively added to a basic tree, corresponding to adding edges to an acyclic graph containing all nodes. The method presented here starts with the whole graph, removing (at least) one edge at a time. Both methods yield a set of independent cycles. If the original full graph satisfies the flow conservation condition, then our method guarantees that the remaining graph still satisfies the condition after each removal of a simple cycle with no contradictory directions, as shown in Corollaries 2.1 and 2.2. Therefore the graph can be completely decomposed into such simple cycles. In applications to life cycle analysis, this property ensures that an elasticity matrix will be decomposed into a complete set of loops with no contradictory directions, as illustrated in Examples 4.1 and 4.2, thereby resolving a standing problem in loop analysis.

The proposed decomposition is generally not unique, just as different spanning trees produce different sets of cycles. The interests of the subject matter should dominate the order and selection of cycles. Important, meaningful cycles should be given higher priority during the decomposition.

The number of simple cycles in a decomposition of a life cycle graph is at most

$$\text{the number of edges} - \text{the number of stages} + 1$$

because the cycles in a decomposition are independent. The complexity of the search procedure for one simple cycle with no contradictory directions is $O(|E(G)|)$, where $|E(G)|$ is the number of the edges of the graph G. For the decomposition of the entire graph into a set of such cycles and a remaining graph, the complexity is $O(|E(G)|^2)$. In other words, the complexity increases rapidly with $|E(G)|$. The algorithm may not be suitable for graphs with very large $|E(G)|$. In practice, we typically look for one set of cycles instead of obtaining all possible decompositions.

Acknowledgements We thank C. Pfister for presenting the challenge of conducting loop analysis for the matrix (5), for kindly allowing us to use her parameter estimates in Example 4.2, and for many conversations that helped us to focus on the biological questions to be answered. We thank the editor for his valuable, constructive suggestions.

6 Appendix

6.1 The decomposition of G in (4)

$$G = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 6.594 \\ 0.025 & 0 & 0 & 0 & 0 & 0 \\ 0.079 & 0.025 & 0.015 & 0 & 0 & 0.152 \\ 0.750 & 0 & 0.256 & 2.773 & 0 & 23.271 \\ 5.740 & 0 & 0 & 19.120 & 2.272 & 4.450 \\ 0 & 0 & 0 & 5.157 & 29.310 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 6.569 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0.079 & 0 & 0.015 & 0 & 0 & 0.152 \\ 0.750 & 0 & 0.231 & 2.773 & 0 & 23.271 \\ 5.740 & 0 & 0 & 19.095 & 2.272 & 4.450 \\ 0 & 0 & 0 & 5.157 & 29.285 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & .025 \\ .025 & 0 & 0 & 0 & 0 & 0 \\ 0 & .025 & 0 & 0 & 0 & 0 \\ 0 & 0 & .025 & 0 & 0 & 0 \\ 0 & 0 & 0 & .025 & 0 & 0 \\ 0 & 0 & 0 & 0 & .025 & 0 \end{bmatrix}$$

$$= G_1 \bigcup L_1 \ (= \{1 \to 2 \to 3 \to 4 \to 5 \to 6 \to 1\})$$

$$G_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 6.490 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.015 & 0 & 0 & 0.152 \\ 0.750 & 0 & 0.152 & 2.773 & 0 & 23.271 \\ 5.740 & 0 & 0 & 19.016 & 2.272 & 4.450 \\ 0 & 0 & 0 & 5.157 & 29.206 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0.079 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0.079 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.079 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.079 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.079 & 0 \end{bmatrix}$$

$$= G_2 \bigcup L_2 \ (= \{1 \to 3 \to 4 \to 5 \to 6 \to 1\})$$

$$G_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 5.740 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.015 & 0 & 0 & 0.152 \\ 0 & 0 & 0.152 & 2.773 & 0 & 23.271 \\ 5.740 & 0 & 0 & 18.266 & 2.272 & 4.450 \\ 0 & 0 & 0 & 5.157 & 28.456 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0.750 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0.750 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.750 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.750 & 0 \end{bmatrix}$$

$$= G_3 \bigcup L_3 \ (= \{1 \to 4 \to 5 \to 6 \to 1\})$$

$$G_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.015 & 0 & 0 & 0.152 \\ 0 & 0 & 0.152 & 2.773 & 0 & 23.271 \\ 0 & 0 & 0 & 18.266 & 2.272 & 4.450 \\ 0 & 0 & 0 & 5.157 & 22.716 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 5.740 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 5.740 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5.740 & 0 \end{bmatrix}$$

$$= G_4 \bigcup L_4 \ (= \{1 \to 5 \to 6 \to 1\})$$

We have exhausted all cycles started from stages **1** and **2**.

$$G_4 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.015 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2.773 & 0 & 23.271 \\ 0 & 0 & 0 & 18.114 & 2.272 & 4.450 \\ 0 & 0 & 0 & 5.157 & 22.564 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.152 \\ 0 & 0 & 0.152 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.152 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.152 & 0 \end{bmatrix}$$

$$= G_5 \bigcup L_5 \ (= \{3 \to 4 \to 5 \to 6 \to 3\})$$

$$G_5 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.015 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2.773 & 0 & 5.157 \\ 0 & 0 & 0 & 0 & 2.272 & 4.450 \\ 0 & 0 & 0 & 5.157 & 4.450 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 18.114 \\ 0 & 0 & 0 & 18.114 & 0 & 0 \\ 0 & 0 & 0 & 0 & 18.114 & 0 \end{bmatrix}$$

$$= G_6 \bigcup L_6 \ (= \{4 \to 5 \to 6 \to 4\})$$

$$G_6 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.015 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2.773 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2.272 & 4.450 \\ 0 & 0 & 0 & 0 & 4.450 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 5.157 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 5.157 & 0 & 0 \end{bmatrix}$$

$$= G_7 \bigcup L_7 \ (= \{4 \to 6 \to 4\})$$

$$G_7 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.015 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2.773 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2.272 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 4.450 \\ 0 & 0 & 0 & 0 & 4.450 & 0 \end{bmatrix}$$

$$= G_8 \bigcup L_8 \ (= \{5 \to 6 \to 5\})$$

where G_8 consists of three self-loops:

$$G_8 = L_9 \bigcup L_{10} \bigcup L_{11} = \{3 \rightleftharpoons 3\} \bigcup \{4 \rightleftharpoons 4\} \bigcup \{5 \rightleftharpoons 5\}$$

Therefore G is decomposed completely into 11 loops:

$$G = \bigcup_{i=1}^{11} L_i$$

6.2 The decomposition of G in (5)

$$G = \begin{bmatrix} 79 & 21 & 2 & 1 & 12 & 10 \\ 34 & 74 & 2 & 2 & 16 & 15 \\ 9 & 14 & 22 & 20 & 0 & 0 \\ 3 & 34 & 14 & 25 & 0 & 0 \\ 0 & 0 & 15 & 12 & 190 & 99 \\ 0 & 0 & 10 & 16 & 98 & 152 \end{bmatrix} = \begin{bmatrix} 79 & 21 & 0 & 1 & 12 & 10 \\ 34 & 74 & 2 & 2 & 16 & 15 \\ 7 & 14 & 22 & 20 & 0 & 0 \\ 3 & 34 & 14 & 25 & 0 & 0 \\ 0 & 0 & 15 & 12 & 190 & 99 \\ 0 & 0 & 10 & 16 & 98 & 152 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$= G_1 \bigcup L_1 (= \{1 \to 3 \to 1\})$$

Subtracting the loop L_1, the remaining graph G_1 can be decomposed as

$$G_1 = \begin{bmatrix} 79 & 21 & 0 & 1 & 12 & 10 \\ 34 & 74 & 2 & 2 & 16 & 15 \\ 7 & 14 & 22 & 20 & 0 & 0 \\ 3 & 34 & 14 & 25 & 0 & 0 \\ 0 & 0 & 15 & 12 & 190 & 99 \\ 0 & 0 & 10 & 16 & 98 & 152 \end{bmatrix} = \begin{bmatrix} 79 & 21 & 0 & 1 & 12 & 10 \\ 34 & 74 & 2 & 0 & 16 & 15 \\ 7 & 14 & 22 & 20 & 0 & 0 \\ 3 & 32 & 14 & 25 & 0 & 0 \\ 0 & 0 & 15 & 12 & 190 & 99 \\ 0 & 0 & 10 & 16 & 98 & 152 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$= G_2 \bigcup L_2 (= \{2 \to 4 \to 2\})$$

Subtracting the loop L_2, the remaining graph G_2 can be decomposed as

$$G_2 = \begin{bmatrix} 79 & 21 & 0 & 1 & 12 & 10 \\ 34 & 74 & 2 & 0 & 16 & 15 \\ 7 & 14 & 22 & 20 & 0 & 0 \\ 3 & 32 & 14 & 25 & 0 & 0 \\ 0 & 0 & 15 & 12 & 190 & 99 \\ 0 & 0 & 10 & 16 & 98 & 152 \end{bmatrix} = \begin{bmatrix} 79 & 21 & 0 & 1 & 5 & 10 \\ 34 & 74 & 2 & 0 & 16 & 15 \\ 0 & 14 & 22 & 20 & 0 & 0 \\ 3 & 32 & 14 & 25 & 0 & 0 \\ 0 & 0 & 8 & 12 & 190 & 99 \\ 0 & 0 & 10 & 16 & 98 & 152 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 7 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 7 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$= G_3 \bigcup L_3 (= \{1 \to 3 \to 5 \to 1\})$$

Subtracting the loop L_3, the remaining graph G_3 can be decomposed as

$$G_3 = \begin{bmatrix} 79 & 21 & 0 & 1 & 5 & 10 \\ 34 & 74 & 2 & 0 & 16 & 15 \\ 0 & 14 & 22 & 20 & 0 & 0 \\ 3 & 32 & 14 & 25 & 0 & 0 \\ 0 & 0 & 8 & 12 & 190 & 99 \\ 0 & 0 & 10 & 16 & 98 & 152 \end{bmatrix} = \begin{bmatrix} 79 & 21 & 0 & 1 & 5 & 10 \\ 34 & 74 & 2 & 0 & 16 & 0 \\ 0 & 14 & 22 & 20 & 0 & 0 \\ 3 & 17 & 14 & 25 & 0 & 0 \\ 0 & 0 & 8 & 12 & 190 & 99 \\ 0 & 0 & 10 & 1 & 98 & 152 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 15 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 15 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 15 & 0 & 0 \end{bmatrix}$$

$$= G_4 \bigcup L_4 (= \{2 \to 4 \to 6 \to 2\})$$

We have obtained the most desirable loops that track slow growers and fast growers. Now subtracting the loop L_4, the remaining graph G_4 can be decomposed in possibly different ways. Consider

$$G_4=\begin{bmatrix}79&21&0&1&5&10\\34&74&2&0&16&0\\0&14&22&20&0&0\\3&17&14&25&0&0\\0&0&8&12&190&99\\0&0&10&1&98&152\end{bmatrix}=\begin{bmatrix}79&21&0&1&5&9\\34&74&2&0&16&0\\0&14&22&20&0&0\\2&17&14&25&0&0\\0&0&8&12&190&99\\0&0&10&0&98&152\end{bmatrix}+\begin{bmatrix}0&0&0&0&0&1\\0&0&0&0&0&0\\0&0&0&0&0&0\\1&0&0&0&0&0\\0&0&0&0&0&0\\0&0&0&1&0&0\end{bmatrix}$$

$$= \quad G_5 \quad \bigcup \; L_5(=\{1\to 4\to 6\to 1\})$$

$$G_5=\begin{bmatrix}79&21&0&1&5&9\\34&74&2&0&16&0\\0&14&22&20&0&0\\2&17&14&25&0&0\\0&0&8&12&190&99\\0&0&10&0&98&152\end{bmatrix}=\begin{bmatrix}79&21&0&1&3&9\\34&74&2&0&16&0\\0&14&22&20&0&0\\0&17&14&25&0&0\\0&0&8&10&190&99\\0&0&10&0&98&152\end{bmatrix}+\begin{bmatrix}0&0&0&0&2&0\\0&0&0&0&0&0\\0&0&0&0&0&0\\2&0&0&0&0&0\\0&0&0&2&0&0\\0&0&0&0&0&0\end{bmatrix}$$

$$= \quad G_6 \quad \bigcup \; L_6(=\{1\to 4\to 5\to 1\})$$

Further,

$$G_6=\begin{bmatrix}79&21&0&1&3&9\\34&74&2&0&16&0\\0&14&22&20&0&0\\0&17&14&25&0&0\\0&0&8&10&190&99\\0&0&10&0&98&152\end{bmatrix}=\begin{bmatrix}79&21&0&1&0&9\\31&74&2&0&16&0\\0&11&22&20&0&0\\0&17&14&25&0&0\\0&0&5&10&190&99\\0&0&10&0&98&152\end{bmatrix}+\begin{bmatrix}0&0&0&0&3&0\\3&0&0&0&0&0\\0&3&0&0&0&0\\0&0&0&0&0&0\\0&0&3&0&0&0\\0&0&0&0&0&0\end{bmatrix}$$

$$= \quad G_7 \quad \bigcup \; L_7(=\{1\to 2\to 3\to 5\to 1\})$$

where

$$G_7=\begin{bmatrix}79&21&0&1&0&9\\31&74&2&0&16&0\\0&11&22&20&0&0\\0&17&14&25&0&0\\0&0&5&10&190&99\\0&0&10&0&98&152\end{bmatrix}=\begin{bmatrix}79&21&0&1&0&0\\22&74&2&0&16&0\\0&2&22&20&0&0\\0&17&14&25&0&0\\0&0&5&10&190&99\\0&0&1&0&98&152\end{bmatrix}+\begin{bmatrix}0&0&0&0&0&9\\9&0&0&0&0&0\\0&9&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&0\\0&0&9&0&0&0\end{bmatrix}$$

$$G_8 \quad \bigcup \; L_8(=\{1\to 2\to 3\to 6\to 1\})$$

$$\Downarrow$$

$$\begin{bmatrix}79&21&0&0&0&0\\21&74&2&0&16&0\\0&2&22&20&0&0\\0&16&14&25&0&0\\0&0&5&10&190&99\\0&0&1&0&98&152\end{bmatrix}+\begin{bmatrix}0&0&0&1&0&0\\1&0&0&0&0&0\\0&0&0&0&0&0\\0&1&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&0\end{bmatrix}$$

$$G_9 \quad \bigcup \; L_9(=\{1\to 2\to 4\to 1\})$$

Now there are no more loops starting at state **1**. Consider

$$G_9 = \begin{bmatrix} 79 & 21 & 0 & 0 & 0 & 0 \\ 21 & 74 & 2 & 0 & 16 & 0 \\ 0 & 2 & 22 & 20 & 0 & 0 \\ 0 & 16 & 14 & 25 & 0 & 0 \\ 0 & 0 & 5 & 10 & 190 & 99 \\ 0 & 0 & 1 & 0 & 98 & 152 \end{bmatrix} = \begin{bmatrix} 79 & 21 & 0 & 0 & 0 & 0 \\ 21 & 74 & 2 & 0 & 14 & 0 \\ 0 & 0 & 22 & 20 & 0 & 0 \\ 0 & 16 & 14 & 25 & 0 & 0 \\ 0 & 0 & 3 & 10 & 190 & 99 \\ 0 & 0 & 1 & 0 & 98 & 152 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$= G_{10} \bigcup L_{10}(= \{2 \to 3 \to 5 \to 2\})$$

$$G_{10} = \begin{bmatrix} 79 & 21 & 0 & 0 & 0 & 0 \\ 21 & 74 & 2 & 0 & 4 & 0 \\ 0 & 0 & 22 & 20 & 0 & 0 \\ 0 & 6 & 14 & 25 & 0 & 0 \\ 0 & 0 & 3 & 0 & 190 & 99 \\ 0 & 0 & 1 & 0 & 98 & 152 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$= G_{11} \bigcup L_{11} = (\{2 \to 4 \to 5 \to 2\})$$

$$G_{11} = \begin{bmatrix} 79 & 21 & 0 & 0 & 0 & 0 \\ 21 & 74 & 2 & 0 & 1 & 0 \\ 0 & 0 & 22 & 17 & 0 & 0 \\ 0 & 3 & 14 & 25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 190 & 99 \\ 0 & 0 & 1 & 0 & 98 & 152 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$= G_{12} \bigcup L_{12}(= \{2 \to 4 \to 3 \to 5 \to 2\})$$

$$G_{12} = \begin{bmatrix} 79 & 21 & 0 & 0 & 0 & 0 \\ 21 & 74 & 2 & 0 & 0 & 0 \\ 0 & 0 & 22 & 16 & 0 & 0 \\ 0 & 2 & 14 & 25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 190 & 98 \\ 0 & 0 & 0 & 0 & 98 & 152 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

$$= G_{13} \bigcup L_{13}(= \{2 \to 4 \to 3 \to 6 \to 5 \to 2\})$$

$$G_{13} = \begin{bmatrix} 79 & 21 & 0 & 0 & 0 & 0 \\ 21 & 74 & 0 & 0 & 0 & 0 \\ 0 & 0 & 22 & 14 & 0 & 0 \\ 0 & 0 & 14 & 25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 190 & 99 \\ 0 & 0 & 0 & 0 & 98 & 152 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$= G_{14} \bigcup L_{14}(= \{2 \to 4 \to 3 \to 2\})$$

We have exhausted all life cycles started from and grown out of stages **1** and **2**. Now

$$G_{14} = \begin{bmatrix} 0 & 21 & 0 & 0 & 0 & 0 \\ 21 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 14 & 0 & 0 \\ 0 & 0 & 14 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 98 \\ 0 & 0 & 0 & 0 & 98 & 0 \end{bmatrix} + \begin{bmatrix} 79 & 0 & 0 & 0 & 0 & 0 \\ 0 & 74 & 0 & 0 & 0 & 0 \\ 0 & 0 & 22 & 0 & 0 & 0 \\ 0 & 0 & 0 & 25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 190 & 0 \\ 0 & 0 & 0 & 0 & 0 & 152 \end{bmatrix}$$

$$= \quad L_{15} \cup L_{16} \cup L_{17} \quad \bigcup \; L_{18} \cup L_{19} \cup L_{20} \cup L_{21} \cup L_{22} \cup L_{23}$$

$$= \; \{1 \rightleftharpoons 2\} \cup \{3 \rightleftharpoons 4\} \cup \{5 \rightleftharpoons 6\} \bigcup \quad \text{6 self loops}$$

Therefore G is decomposed completely into 23 loops:

$$G = \bigcup_{i=1}^{23} L_i$$

References

1. Bang-Jensen, J., Gutin, G.: Digraphs: theory, algorithms and applications. Springer, Berlin Heidelberg New York (2001)
2. Carré, B.: Graphs and Networks. Clarendon Press, Oxford (1979)
3. Caswell, H.: Matrix Population Models: construction, analysis, and interpretation, 2nd edn. Sinauer, Sunderland (2001)
4. Caswell, H., Werner, P.A.: Transient behavior and life history analysis of teasel (Dipsacus sylvestris Huds.). Ecology **59**, 53–66 (1978)
5. Diestel, R.: Graphs theory, 2nd edn. Springer, Berlin Heidelberg New York (2000)
6. van Groenendael, J., de Kroon, H., Kalisz, S., Tuljapurkar, S.: Loop analysis: evaluating life history pathways in population projection matrices. Ecology **75**, 2410–2415 (1994)
7. Jungnickel, D.: Graphs, networks and Algorithms. Springer, Berlin Heidelberg New York (1999)
8. Pfister, C.A., Stevens, F.R.: Individual variation and environmental stochasticity: implications for matrix model predictions. Ecology **84**, 496–510 (2003)
9. Pfister, C.A., Wang, M.: Beyond size: matrix projection models for populations where size is an incomplete descriptor. Ecology **86**, 2673–2683 (2005)
10. Shea, K., Rees, M., Wood, S.N.: Trade-offs, elasticities and the comparative method. J. Ecol. **82**, 951–957 (1994)
11. Silvertown, J., Franco, M., McConway, K.: A demographic interpretation of Grime's triangle. Funct. Ecol. **6**, 130–136 (1992)
12. van Tienderen, P.H.: Life cycle trade-offs in matrix population models. Ecology **76**(8), 2482–2489 (1995)
13. Wardle, G.M.: A graph theory approach to demographic loop analysis. Ecology **79**(7), 2539–2549 (1998)
14. Werner, P.A.: Predictions of fate rosette size in teasel (Dipsacus fullonum L.). Oecologia **20**, 197–201 (1975)
15. Werner, P.A., Caswell, H.: Population growth rates and age vs. stage distribution models for teasel (Dipsacus sylvestris Huds.). Ecology **58**, 1103–1111 (1977)