NUMERICAL LINEAR ALGEBRA WITH APPLICATIONS
Numer. Linear Algebra Appl. 2001; **8**:381–399 (DOI: 10.1002/nla.253)

Computation and presentation of graphs displaying closure hierarchies of Jordan and Kronecker structures

Erik Elmroth, Pedher Johansson and Bo Kågström*

Department of Computing Science, Umeå University, SE-901 87 Umeå, Sweden

SUMMARY

StratiGraph, a Java-based tool for computation and presentation of closure hierarchies of Jordan and Kronecker structures is presented. The tool is based on recent theoretical results on stratifications of orbits and bundles of matrices and matrix pencils. A stratification reveals the complete hierarchy of nearby structures, information critical for explaining the qualitative behaviour of linear systems under perturbations. StratiGraph facilitates the application of these theories and visualizes the resulting hierarchy as a graph. Nodes in the graph represent orbits or bundles of matrices or matrix pencils. Edges represent covering relations in the closure hierarchy. Given a Jordan or Kronecker structure, a user can obtain the complete information of nearby structures simply by mouse clicks on nodes of interest. This contribution gives an overview of the StratiGraph tool, presents its main functionalities and other features, and illustrates its use by sample applications. Copyright © 2001 John Wiley & Sons, Ltd.

KEY WORDS: stratification; Jordan and Kronecker canonical form; structure hierarchy; orbit; bundle; StratiGraph; controllability; observability; matrix pencils; perturbation theory

1. INTRODUCTION

It is well known that computing the fine canonical structure elements of matrices and linear matrix pencils are ill-posed problems. Arbitrary small perturbations in the data may drastically change the canonical structure. For example, close or otherwise ill-conditioned eigenvalues may coalesce, or a singular pencil becomes regular. Let us consider the following 3×3 matrix pencils:

$$\left[\begin{array}{c|c|c} 1-\lambda & 0 & 0 \\ \hline 0 & \delta-\lambda & 0 \\ \hline 0 & 0 & -\lambda \end{array}\right], \quad \left[\begin{array}{c|c|c} 1 & 0 & 0 \\ \hline 0 & 0 & 0 \\ \hline 0 & 0 & -\lambda \end{array}\right], \quad \left[\begin{array}{cc|c} 1 & -\lambda & 0 \\ \hline 0 & 0 & 1 \\ 0 & 0 & -\lambda \end{array}\right]$$

which we denote $A_1-\lambda B_1$, $A_2-\lambda B_2$ and $A_3-\lambda B_3$, respectively. They all live in the $18(=2\cdot 3^2)$-dimensional space of 3×3 matrix pencils. Since $\det(A_1-\lambda B_1)=0$ only for $\lambda=1,\delta$ and 0,

*Correspondence to: Bo Kågström, Department of Computing Science, Umeå University, SE-901 87 Umeå, Sweden

Contract/grant sponsor: Swedish Research Council for Engineering Sciences; contract/grant number: TFR 222-97-112.

Received 25 April 2000
Revised 25 August 2000

the first pencil is regular with three distinct eigenvalues. Moreover, it is within an $O(\delta)$ distance from a pencil with a multiple eigenvalue at 0. Obviously, $\det(A_2-\lambda B_2)\equiv 0$ and $\det(A_3-\lambda B_3)\equiv 0$ for all λ, i.e. they are singular pencils. Although, both pencils have the same diagonal elements they have very different canonical forms, which their different block structures above depict. Before we have introduced the notation for the Jordan and Kronecker canonical forms, we only mention here that $A_2-\lambda B_2$ has two eigenvalues (at zero and infinity) and a singular part of size 1×1, while $A_3-\lambda B_3$ is a singular pencil with no eigenvalues. By adding arbitrary small perturbations to the $(2,2)$ entries of $A_2-\lambda B_2$ and $A_3-\lambda B_3$, both pencils become regular. But what is the relation between these three pencils within the 18-dimensional space of 3×3 matrix pencils?

All these cases and similar examples may show up in different ill-posed control applications [1, 2]. Besides knowing the canonical structure of a system pencil associated with a linear system, it is equally important to know what are the nearby canonical structures that explain the behaviour of the system under small perturbations.

In this contribution, we present StratiGraph, a tool for computing and presenting such qualitative information for general $n\times n$ matrices and $m\times n$ matrix pencils. While stratigraphy is the key to understanding the geological evolution of the world [3], StratiGraph is the key to understanding the 'geometrical evolution' of orbits and bundles in the 'world' of matrices and matrix pencils.

The underlying theory, known as the stratification of orbits and bundles of matrices and matrix pencils, was recently presented [4]. The application of this theory is, however, fairly complicated and requires both good theoretical understanding and tedious and careful work when applied to a real problem. The main motivation for the StratiGraph tool is to facilitate the application of the stratification theories and to provide the user with an easy to grasp representation of the result. Our goal is to present the groundwork for practising StratiGraphers.

StratiGraph computes a connected graph where the nodes correspond to different canonical structures. The graph is a representation of the *stratification* of the possible canonical forms and shows which structures are near other structures (in the sense of being in the closure) in the space of matrices or matrix pencils.

In Plate 1, the stratification or the closure hierarchy graph for the equivalence bundles of the set of 3×3 matrix pencils is presented. Notation and definition of concepts relating to this graph is presented in Section 2, including bundles and closure relations. Here, we just mention that $A_1-\lambda B_1$ corresponds to the top-most node, $A_2-\lambda B_2$ and $A_3-\lambda B_3$ belong to the nodes labeled '6 over 1' and '4 over 3', respectively.

The set of pencils equivalent to $A_1-\lambda B_1$ spans the space of 3×3 matrix pencils. This corresponds to the generic case, i.e. almost all pencils are diagonalizable and have distinct eigenvalues. All nodes below the top node in the graph represent spaces of dimension less than 18, and they all correspond to degenerate canonical structures. The bottom-most node corresponds to the 3×3 pencil with zero entries, which has the most degenerate structure. We return to these examples and Plate 1 later.

Before, we go into further details we outline the rest of the paper. Section 2 introduces the necessary definitions and notation for the matrix and matrix pencil spaces, including canonical forms and structure characteristics, orbits and bundles, genericity and codimension, the concepts of stratifications and covering relations. In Section 3, we give an overview of the StratiGraph tool and its functionalities. Section 4 presents the theory behind the algorithms implemented in StratiGraph for computing the stratifications for orbits and bundles of matrices

and matrix pencils. The algorithms are based on recent results by Edelman, Elmroth and Kågström [4]. Nearest neighbours, i.e. structures satisfying covering relations with a specified structure, are identified in terms of coin moves in coin tables. Each coin table corresponds to an integer partition representing one subset of the structure characteristics for a specified canonical structure. In Section 5, we describe some software design principles that have been used in the implementation of the Java-based tools. StratiGraph is fully compatible with Java 1.1. Section 6 illustrates the use of StratiGraph with a few examples, including its use in a control application. Finally, in Sections 7 and 8, we report on related work and give information about the availability of the software and some possible extensions to our tools.

2. THE MATRIX AND MATRIX PENCIL SPACES

2.1. *Orbits, bundles, canonical forms and structure characteristics*

Any $n \times n$ matrix A defines a manifold of similar matrices in the n^2-dimensional space $\mathscr{M}$ of square matrices. This manifold is the *similarity orbit* defined as

$$\mathscr{O}(A) = \{P^{-1}AP:\ \det(P) \neq 0\}$$

We may choose a special element of $\mathscr{O}(A)$ that reveals the *Jordan canonical form* (JCF) of the matrix [5]:

$$AP = PJ$$

where

$$J = \operatorname{diag}\{J(\lambda_1), J(\lambda_2), \ldots, J(\lambda_t)\}$$

Here, we assume that A has t distinct eigenvalues λ_i with algebraic multiplicities a_i. The Jordan matrix $J(\lambda_i)$ is a direct sum of the Jordan blocks associated with the eigenvalue λ_i. The number of Jordan blocks for an eigenvalue is the same as its geometric multiplicity g_i. Let $s_k^{(i)}$ be the sizes of the Jordan blocks associated with λ_i, where $s_1^{(i)} \geqslant s_2^{(i)} \geqslant \cdots \geqslant s_{g_i}^{(i)} \geqslant 1$. Algebraically, the $s_k^{(i)}$s are the degrees of the elementary divisors of $A - \lambda I$ at $\lambda = \lambda_i$, also known as the *Segre characteristics*. We also see that $h_i = s_1^{(i)}$ is the maximum height of the vector chains for the eigenvalue λ_i. The Jordan matrix $J(\lambda_i)$ can be expressed as

$$J(\lambda_i) = \operatorname{diag}\{J_{s_1^{(i)}}(\lambda_i), J_{s_2^{(i)}}(\lambda_i), \ldots, J_{s_{g_i}^{(i)}}(\lambda_i)\}$$

The complete Jordan matrix J is uniquely defined up to the order of the Jordan blocks.

We also mention a dual way to characterize the JCF. Let $w_k^{(i)}$ be the number of principal vectors of grade k associated with λ_i (or equivalently the number of $J_j(\lambda_i)$ blocks of size $j \geqslant k$). We have that $w_1^{(i)} \geqslant w_2^{(i)} \geqslant \cdots \geqslant w_{h_i}^{(i)} \geqslant 1$, which is the set of nonzero successive differences in the nullities of the matrices $(A - \lambda_i I)^k$ for $k = 1, \ldots, h_i$, also known as the *Weyr characteristics*.

The matrix pencil analogue is to consider any $m \times n$ matrix pair (A, B). Then $A - \lambda B$ defines an *orbit of strictly equivalent matrix pencils* in the $2mn$-dimensional space $\mathscr{P}$ of $m \times n$ pencils:

$$\mathscr{O}(A - \lambda B) = \{P^{-1}(A - \lambda B)Q:\ \det(P)\det(Q) \neq 0\}$$

Similar to the matrix case, we may choose a special element of $\mathcal{O}(A-\lambda B)$ that exhibits the fine structure blocks of the matrix pencil, namely the *Kronecker canonical form* (KCF) [5]. In addition to Jordan blocks for finite and infinite eigenvalues, the Kronecker form contains singular blocks corresponding to minimal indices of a singular pencil.

Suppose $A, B \in \mathbf{C}^{m\times n}$. Then there exist non-singular $P \in \mathbf{C}^{m\times m}$ and $Q \in \mathbf{C}^{n\times n}$ such that

$$P^{-1}(A-\lambda B)Q = \tilde{A}-\lambda\tilde{B} \equiv \operatorname{diag}(A_1-\lambda B_1, \ldots, A_b-\lambda B_b)$$

where $A_i-\lambda B_i$ is $m_i \times n_i$. Each block $M_i \equiv A_i-\lambda B_i$ must be of one of the following forms:

$$J_j(\alpha), \quad N_j, \quad L_j, \quad \text{or} \quad L_j^{\mathrm{T}}$$

The $J_j(\alpha)$ and N_j are simply Jordan blocks of the finite and infinite eigenvalues, respectively. These blocks together constitute the *regular structure* of the pencil.

The other two types of diagonal blocks are

$$L_j \equiv \begin{bmatrix} -\lambda & 1 & & \\ & \ddots & \ddots & \\ & & -\lambda & 1 \end{bmatrix} \quad \text{and} \quad L_j^{\mathrm{T}} \equiv \begin{bmatrix} -\lambda & & \\ 1 & \ddots & \\ & \ddots & -\lambda \\ & & 1 \end{bmatrix}$$

The $j \times (j+1)$ block L_j is called a *singular block of right* (*or column*) *minimal index* j. It has a one dimensional right nullspace, $r_j = [1, \lambda, \ldots, \lambda^j]^{\mathrm{T}}$, such that $L_j r_j = 0$ for all λ. Similarly, the $(j+1) \times j$ block L_j^{T} is a *singular block of left* (*or row*) *minimal index* j, and has a one dimensional left nullspace for any λ. The left and right singular blocks together constitute the *singular structure* of the pencil, and appear in the KCF if and only if the pencil is singular. The regular and singular structures define the *Kronecker structure* of a singular pencil. Also, the KCF is uniquely defined up to the order of the canonical blocks. For consistency reasons, the L_j blocks appear before the L_j^{T} blocks, while the Jordan blocks of the regular part may appear anywhere along the block diagonal of $\tilde{A}-\lambda\tilde{B}$.

Besides the KCF block notation, we can also introduce Segre and Weyr characteristics. These characteristics for the Jordan structure of finite and infinite eigenvalues follow the matrix case and we let $\mathcal{J}_\mu(A-\lambda B)$ denote the integer partition for the Jordan structure corresponding to the eigenvalue μ (finite or infinite). Moreover, let $\mathcal{R}(A-\lambda B)$ and $\mathcal{L}(A-\lambda B)$ denote the partitions for the right and left structures, respectively. When it is clear from context, we use the abbreviated notation $\mathcal{R}$, $\mathcal{L}$, and $\mathcal{J}_\mu$.

In the following, we let these partitions correspond to the *Weyr characteristics* of $A-\lambda B$. The j_k's in a $\mathcal{J}_\mu$ partition are the Weyr characteristics for the eigenvalue μ, i.e., j_k is the number of $J_j(\mu)$ blocks of size $j \geqslant k$. Similarly, r_k of $\mathcal{R}$ is the number of L_j blocks of size $j \geqslant k$, and l_k of $\mathcal{L}$ is the number of L_j^{T} blocks of size $j \geqslant k$. We illustrate the notation with the KCF

$$\operatorname{diag}(L_0, L_2, J_1(\mu_1), J_2(\mu_1), J_3(\mu_2), L_1^{\mathrm{T}})$$

which corresponds to the Weyr structure partitions

$$\mathcal{R} = (2,1,1), \quad \mathcal{L} = (1,1), \quad \mathcal{J}_{\mu_1} = (2,1), \quad \mathcal{J}_{\mu_2} = (1,1,1)$$

We append zero entries to these arrays when it is necessary to consider partitions of equal lengths, including infinite sequences. A special case is an empty partition, which also can be represented as an infinite sequence with only zero entries, e.g. $\mathscr{L} = (0, 0, \ldots)$.

Two elements of an orbit have exactly the same canonical structure, including the types and sizes of the blocks and the eigenvalues. A *bundle* is a union of orbits all with the same canonical structure, but their eigenvalues may differ. The concept of bundle is defined both for matrices and matrix pencils and we use the notation $\mathscr{B}(A)$ and $\mathscr{B}(A-\lambda B)$, respectively,

2.2. Generic and degenerate Kronecker structures

The KCF looks quite complicated in the general case, but most matrix pencils have a more simple Kronecker structure. Almost all rectangular $m \times n$ pencils $A-\lambda B$ ($m \neq n$) have the same KCF, depending only on m and n and this KCF only includes L_j (if $m < n$) and L_j^{T} blocks otherwise (e.g. see References [6,7,8]). This corresponds to the *generic case* when $A-\lambda B$ has full rank for any scalar λ. It follows that generic rectangular pencils have no regular part. We remark that any pencil with only L_j or only L_j^{T} blocks has full rank, but it is only one of them which is the generic structure.

Square pencils are generically regular, i.e. $\det(A-\lambda B) = 0$ if and only if λ is an eigenvalue. Moreover, the most generic regular pencil is diagonalizable with distinct finite eigenvalues. The generic singular pencils of size $n \times n$ have the Kronecker structures [9]:

$$\mathrm{diag}(L_j, L_{n-j-1}^{\mathrm{T}}), \quad j = 0, \ldots, n-1$$

Only if a singular $A-\lambda B$ is rank deficient (for some λ), the associated KCF may be more complicated and include a regular part, as well as, right and left singular blocks. This situation corresponds to a *degenerate* (or *non-generic*) *case*, which is the real challenge from a computational point of view [10, 11]. Degenerate rectangular pencils have several applications in control theory, for example, computing the controllable subspace and the uncontrollable modes of a linear descriptor system [1, 2].

It is now obvious that $A_1-\lambda B_1$, considered in Section 1, is a generic regular pencil. Both singular pencils are in KCF; $A_2-\lambda B_2 = \mathrm{diag}(N_1, L_0, L_0^{\mathrm{T}}, J_1(0))$, and $A_3-\lambda B_3 = \mathrm{diag}(L_1, L_1^{\mathrm{T}})$. From top to bottom, the diagonal blocks of $A_2-\lambda B_2$ correspond to N_1, $\mathrm{diag}(L_0, L_0^{\mathrm{T}})$ and J_1. So $A_2-\lambda B_2$ has a regular part of size 2×2 with eigenvalues at zero ($J_1(0)$) and infinity (N_1) and a singular part of size 1×1 corresponding to one L_0 block (of size 0×1) and one L_0^{T} block, i.e., the pencil is degenerate. $A_3-\lambda B_3$ has no regular part and is a generic 3×3 singular pencil.

2.3. Closure hierarchies of Jordan and Kronecker structures

Both orbits and bundles are manifolds with well-defined dimensions. The dimension of an orbit is equal to the dimension of its tangent space. The normal space of an orbit at a certain point (matrix or matrix pencil) is the space complementary and orthogonal to the tangent space at this point. The dimension of the normal space is the *codimension* ($\mathrm{cod}(\cdot)$) of the orbit, which depends only on the Jordan and Kronecker structures (e.g. see Reference [7]). The codimension can also be computed as the number of zero singular values of a block matrix of Kronecker products [8].

If we do not wish to specify the value of an eigenvalue, which corresponds to the bundle case, the codimension count for this unspecified eigenvalue is one less. Therefore,

$$\operatorname{cod}(\mathscr{B}(A-\lambda B)) = \operatorname{cod}(\mathscr{O}(A-\lambda B)) - \text{the number of distinct eigenvalues of } A-\lambda B$$

A similar expression holds for the matrix case.

For the orbit of a generic $n \times n$ matrix or matrix pencil we have $\operatorname{cod}(\cdot) = n$ (n specified and distinct eigenvalues). The orbit of a generic rectangular matrix pencil spans the complete space, i.e., $\operatorname{cod}(\cdot) = 0$. At the other extreme, we have the least generic, i.e. the *most degenerate* cases. These are the zero matrix $A = 0_{n\times n}$ and the zero pencil $A-\lambda B = 0_{m\times n} - \lambda 0_{m\times n}$. The corresponding orbits have $\operatorname{cod}(A) = n^2$ and $\operatorname{cod}(A-\lambda B) = 2mn$. All other (degenerate) cases fall between these extremes; the *generic canonical forms* describe almost all matrices and pencils and the 'null forms' describe only one single matrix and pencil. They are points in n^2- and $2mn$-spaces, respectively.

These facts induce a natural hierarchy; one matrix or pencil is more generic than another if it has lower codimension. However, this classification does not give the complete picture. Orbits and bundles of different canonical structures of a given size can have the same codimensions. (See examples in Plate 1, where nodes on the same level in the graph represent bundles with the same codimension). We are also interested to know the relations between these and the other structures above and below in the hierarchy. The answer is given by a *stratified manifold*, which is the union of nonintersecting manifolds whose closure is the finite union of itself with strata of smaller dimensions (thereby defining stratified manifolds recursively, see Arnold [12]). For matrices, the strata is similarity orbits or bundles. For pencils, the strata is equivalence orbits or bundles. We use the notation $\bar{\mathscr{O}}(\cdot)$ and $\bar{\mathscr{B}}(\cdot)$ to denote the closures of orbits and bundles, respectively.

The *problem of stratification* is to find the closure relations among the various orbits or bundles, i.e., the closure hierarchy of Jordan and Kronecker structures. These relations define a partial ordering on orbits or bundles, which we name a *covering relationship*. One structure covers another if its closure includes the closure of the other and there is no other structure in between.

The stratification is a classification of all possible changes in the canonical structure that can take place for sufficiently small perturbations of a given matrix or pencil. Moreover, every possible change is smoothly attainable in terms of versal deformations (e.g. see Arnold [12] for the matrix case and Edelman, Elmroth and Kågström [8] for general matrix pencils).

3. THE STRATIGRAPH TOOL

StratiGraph is a tool that computes and displays the stratification of orbits and bundles of Jordan and Kronecker structures as a connected graph with nodes and edges. This section is devoted to an overview of this tool. First we illustrate how a stratification is presented in StratiGraph. We continue with support for analysing a stratification and how the user can interact with the tool. Finally, we present some additional features.

3.1. The StratiGraph representation of a stratification

StratiGraph represents a stratification by a graph in the following way. A node represents a space defined as an orbit or a bundle of matrices or matrix pencils. An edge represents a

covering relation between two orbits or bundles, so that the one above in the graph covers the one below. The definition of covering relations is given in Section 2.3. It follows that all orbits or bundles that can be reached by downward paths from a given node define orbits or bundles that are in the closure of the one represented by this node.

We now illustrate this representation by considering Plate 1, where StratiGraph presents the complete stratification of bundles of matrix pencils of size 3×3. These pencils live in a 18-dimensional space.

The codimensions are given on the left hand side of the window. The topmost node, marked '0 over 1' corresponds to the generic 3×3 pencil with codimension 0. This means that the closure of this bundle is the complete 18-dimensional space. All other structures are in the closure of this bundle. In the graph, this is represented by paths from the generic structure to all other structures below. (For now, we only focus on the structure of the graph and what it represents in the 18-dimensional space, and do not bother which Kronecker structure is represented by each node.)

The second most generic structure, with codimension 1, is represented by the node '1 over 1' immediately below the generic structure. This node represents an 17-dimensional space, as the codimension is 1. The edge from the generic structure illustrates that there is a covering relation between the two structures.

Downwards from this node, we find two edges. Hence, the structure represented by this node covers two other nodes. The node '2 over 1' represents a 16- and '3 over 1' a 15-dimensional space. Both spaces are subspaces of the closure of the 17-dimensional space, but none of them are in the closure of the other.

We find four different bundles with codimension 4. The node '4 over 1' differs from the other three in that is has two upward edges, i.e. the space represented by this node is in the closure of two other spaces. In fact, this is the 14-dimensional intersection between the closure of the 16-dimensional '2 over 1' and the closure of the 15-dimensional '3 over 1'.

As we follow the paths downwards, we find '12 over 1' as a six-dimensional space in the intersection of the respective closures of three other spaces. Finally, at codimension 18, we find the most degenerate structure.

With codimension 18, this bundle is only a point in the 18-dimensional space ($A = B = 0$). Since all other structures can be found by following paths upward from this bundle, the zero pencil is in the closure of every other bundle. This is of course also easy to verify, since any Kronecker structure can be imposed by a small perturbation of the zero pencil.

3.2. StratiGraph basic functionalities

In addition to the graph, the StratiGraph main window includes some other information. In Plate 1, the circled numbers 1–8 indicate the following:

① Toolbar with buttons for frequently used commands.
② Window displaying the canonical structure corresponding to the active node or edge.
③ Codimension of the structures on this level of the graph.
④ Graph view area.
⑤ Status field that shows some messages to the user.
⑥ Type information of the visible graph: orbits or bundles, matrices or pencils and their sizes.

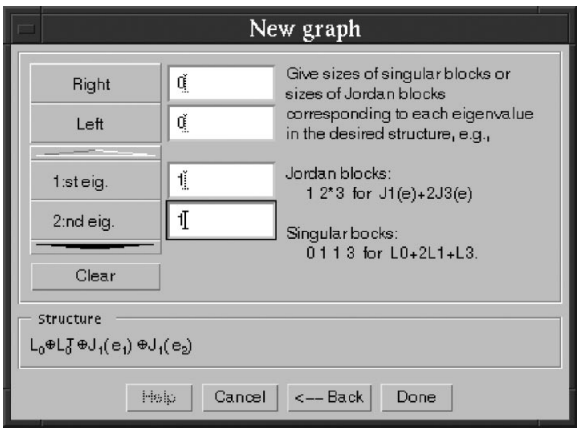


Figure 1. The second page of the 'New graph' dialogue window, allows the user to specify the sizes of individual blocks in the initial structure.

⑦ Shows the notation used to present the structure information (block, Weyr or Segre).
⑧ Covering structures window.

When analysing a stratification, user interaction and further information about the nodes in the graph are important. The numbers given in the nodes, for example, '4 over 2' act as identifiers, that are used to associate the node with additional information. Here, the '4' is the codimension and the '2' is simply a number used to distinguish nodes with the same codimension.

The graph is created by the user. Initially, the user specifies a structure as a starting point, either by giving the explicit Jordan or Kronecker structure or by asking for the most or least generic structure of a specified size. Then the graph is expanded upwards and/or downwards by mouse clicks. A new *current node* is selected by a mouse click and the graph is further expanded upwards or downwards by an additional mouse click on the upper or lower part of that node. The user can also choose to recursively expand the complete subgraph above or below the current node. During this process, StratiGraph regularly asks the user if the recursive process should proceed, so that the same functionality also can be used to expand parts of that subgraph.

For large matrices or matrix pencils, the user is normally not interested in the complete stratification, but only of the stratification in the near vicinity of the initial structure. When specifying the initial matrix or matrix pencil structure, the user also needs to specify if the stratification should be of orbits or bundles. The dialogue window used for creating a new graph is presented in Figure 1.

Information about which structure that is represented by each node in the graph is, of course, absolutely vital for the user. StratiGraph presents this information, in the *Generated structures* window in Plate 2.

This window contains the structure information for all nodes generated. The user may choose from three different notations. On the left hand side of Plate 2, we illustrate the *Block notation* which is illustrative and a natural choice for a small introductory example, but often impractical for large problems. In the middle of the figure, we have the the *Segre notation*,

which is a straightforward way of compressing the block notation. The *Weyr notation*, on the right hand side, is more natural from a staircase algorithm point of view.

StratiGraph provides extra support for identifying covering relations. The *Covered structures* window is the small window marked '8' in Plate 1. It shows only the current structure and the structures covering and covered by the current structure. The notation used is the same as for the Generated structures window.

In all three windows (Main, Generated structures, and Covering structures), the current structure is marked red and the structures fulfiling covering relations are marked in blue. Moreover, the covered and covering structures are marked by arrows pointing downward and upward, respectively, in the Generated structures and Covering structures windows.

For large structures, it may also be difficult to identify the actual differences between two covering structures. Therefore, StratiGraph includes the functionality to display the differences between to nodes in the graph. The function is invoked by a double mouse click on the edge connecting the two nodes.

3.3. Additional features

StratiGraph also provides some basic features for facilitating work with stratifications. This includes functions for printing the stratification graph and associated information to a file or a printer, save a graph representation, and to open a saved graph. The user can also control the zooming of the graph and the font size used for text. In order to focus on a specific part of the graph, to highlight a specific relation, or simply to organize the graph to the liking of the user, StratiGraph allows for the user to rearrange the graph manually. Of course, the user may only move nodes horizontally.

All functions are available from menus. Some are also available as shortcut buttons on the main toolbar, and some can be activated simply by mouse clicks. For detailed information about all functions, menus, shortcuts etc., and more specific information about how to work with this tool, we refer to the StratiGraph User's Guide [27].

4. STRATIFICATION ALGORITHMS IN STRATIGRAPH

In the StratiGraph tool, a key functionality is to determine all Jordan or Kronecker structures that cover a given (specified) structure or that are covered by a given structure. In this section, we summarize the theory and algorithms behind these StratiGraph functions. The theory constitutes a set of rules expressed in terms of what changes need to be imposed on a matrix or a matrix pencil in order to find covering structures, i.e. neighbours in the orbit or bundle stratification.

The rules for finding a matrix or a matrix pencil whose orbit or bundle is in the closure of a given orbit or bundle are presented in Reference [4]. In the graph representation of a stratification, this corresponds to the rules for finding (or generating) nodes immediately below a given node. In the StratiGraph tool, we have implemented functionality to expand a graph upwards and downwards. Therefore, we have extended these rules with the explicit rules for finding the nodes above a given node in the graph.

The rules may be used to determine a complete stratification. Starting from a generic structure of a specified size, this is done by repeatedly applying the rules for finding the next more non-generic structures in the stratification, i.e. structures covered by the current structure.

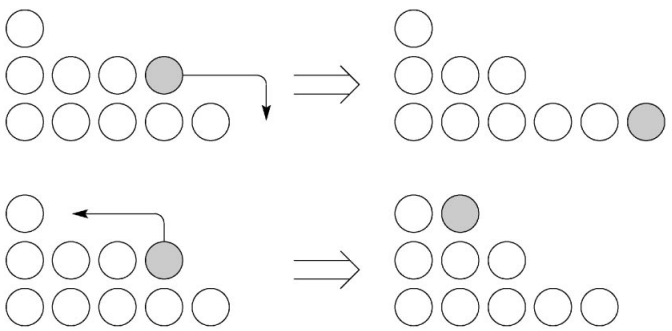


Figure 2. Minimum rightward and leftward 'coin moves' illustrate that $(3,2,2,2,1)$ covers $(3,2,2,1,1,1)$ and $(3,2,2,2,1)$ is covered by $(3,3,2,1,1)$, respectively.

When only the least generic (most degenerate) structure remains, the complete stratification is determined. Similarly, the complete stratification can also be determined by repeatedly applying rules for the next more generic structures, starting from the least generic one.

Another use of the rules is for determining which structures may be in the near vicinity of a given (specified) structure, for example, a structure computed using the GUPTRI software [10, 11, 13]. This can be done by applying rules for finding both more and less generic structures.

All rules are expressed in terms of changes in integer partitions ($\mathscr{J}_{\mu_i}$ for all eigenvalues μ_i, $\mathscr{R}$, and $\mathscr{L}$). Following Reference [4], we identify structures satisfying covering relations with a specified structure, by interpreting the integer partitions as coin tables, and changes in the partitions in terms of coin moves.

We represent a partition $\mathscr{W} = \{w_1, w_2, \ldots, w_k\}$ as a coin table, with k columns of coins with w_i coins in the ith column. A covering relationship between two integer partitions can easily be interpreted as a minimal coin move. (See Reference [14], for an introduction to dominance orderings of integer partitions.) A partition $\mathscr{W}_1$ covers a partition $\mathscr{W}_2$ *if and only if* $\mathscr{W}_2$ can be obtained from $\mathscr{W}_1$ by a minimal rightward coin move. A *minimal rightward coin move* is moving one coin rightward one column or downward one row, keeping the monotonicity requirement. A *minimum leftward coin move* is moving one coin leftward one column or upward one row, also subject to the monotonicity constraint. Minimum rightward and leftward coin moves and the corresponding covering relations are illustrated in Figure 2.

Given a matrix A and the Weyr characteristics represented by an integer partition $\mathscr{J}_{\mu_i}(A)$ for each eigenvalue μ_i of A, we can find a matrix $\tilde{A}$ immediately under or above A in the orbit and bundle stratifications by applying the rules in Table I.

The rules on the left hand side of Table I are for finding a matrix $\tilde{A}$ whose orbit or bundle is covered by the one of A. In the graph representation of a stratification, this corresponds to a node below A's node, i.e. the rules are used for expanding the graph downwards. The rules on the right hand side are for expanding the graph upwards, i.e. for finding $\tilde{A}$ above A in the stratification graph.

In the stratification of matrix orbits, all changes are made within individual integer partitions, since eigenvalues may never coalesce. Consequently, to understand this problem, it is sufficient to study a case with all eigenvalues equal, e.g. the nilpotent case.

Table I. Given the integer partitions of A, the following if-and-only-if rules find $\tilde{A}$ fulfiling orbit or bundle covering relations with A.

$\mathcal{O}(A)$ covers $\mathcal{O}(\tilde{A})$:	$\mathcal{O}(A)$ is covered by $\mathcal{O}(\tilde{A})$:
(1) Minimum *leftward* coin move in any $\mathcal{J}_{\mu_i}$.	(1) Minimum *rightward* coin move in any $\mathcal{J}_{\mu_i}$.
$\mathcal{B}(A)$ covers $\mathcal{B}(\tilde{A})$:	$\mathcal{B}(A)$ is covered by $\mathcal{B}(\tilde{A})$:
(1) Minimum *leftward* coin move in any $\mathcal{J}_{\mu_i}$.	(1) Minimum *rightward* coin move in any $\mathcal{J}_{\mu_i}$.
(2) Let any pair of eigenvalues coalesce, i.e. take the *union* of their sets of coins.	(2) For any eigenvalue μ_i, *divide* the set of coins into two partitions so that their union is $\mathcal{J}_{\mu_i}$.

For matrix bundles, we must also allow for changes in eigenvalues, i.e. two different eigenvalues may coalesce and thereby give a less generic Jordan structure. In addition to left and right minimum coin moves we here also need to define the *union* of two integer partitions (two eigenvalues coalesce). This is a new partition consisting of all the columns of the two partitions ordered according to the monotonicity constraint. The operation contrary to taking the union is the *divide* operation, i.e. to distribute the columns of a partition into two partitions so that their union is the original partition.

We now turn to the matrix pencil case. Since the Kronecker canonical form includes Jordan blocks as well as left and right singular blocks, we now have to consider the integer partitions $\mathcal{R}$, $\mathcal{L}$, and $\mathcal{J}_{\mu_i}$ for all eigenvalues μ_i. Given $A-\lambda B$ we find $\tilde{A} - \lambda\tilde{B}$ immediately under or above $A-\lambda B$ in the orbit and bundle stratifications by applying the rules in Table II.

In contrast to the corresponding matrix orbit case, it is not sufficient to consider only one eigenvalue for the matrix pencil orbit case, even though it is not allowed for two eigenvalues to coalesce.

The rules for finding a pencil whose orbit is covered by the one of $A-\lambda B$ is carefully explained in Reference [4]. When 'reversing' these rules for finding a pencil whose orbit is covering $A - \lambda B$ some attention should be given to the following. In rule 2 (on the right hand side of Table II), the rightmost coin removed from a $\mathcal{J}_{\mu_i}$ may be the last coin in that partition, hence the rule may make an eigenvalue disappear. The actual effect of rule 4 is that the largest left and right singular blocks are replaced by a number of Jordan blocks. For each eigenvalue, a new Jordan block of at least the size of the previously largest block is created. If the total size of the two singular blocks are larger than that, then some Jordan blocks become even larger and/or Jordan blocks for new eigenvalues are created. As a consequence, if the two largest singular blocks are large compared to the total size of the largest Jordan blocks, there are many valid combinations, leading to many structures that cover the specified one.

Finally, the nearest neighbours below and above a given matrix pencil in the matrix pencil bundle stratification are given by the lower half of Table II.

Again, we explain the more critical steps in the rules for finding a pencil above in the hierarchy (on the right hand side of Table II). The two restrictions of rule 4 may need some explanation. First, the rule may be applied to create one row of coins if there are no Jordan

Table II. Given the integer partitions of $A-\lambda B$, the following if-and-only-if rules find $\tilde{A}-\lambda\tilde{B}$ fulfiling orbit or bundle covering relations with $A-\lambda B$.

$\mathcal{O}(A-\lambda B)$ covers $\mathcal{O}(\tilde{A}-\lambda\tilde{B})$:	$\mathcal{O}(A-\lambda B)$ is covered by $\mathcal{O}(\tilde{A}-\lambda\tilde{B})$:
(1) Minimum *rightward* coin move in $\mathscr{R}$ (or $\mathscr{L}$).	(1) Minimum *leftward* coin move in $\mathscr{R}$ (or $\mathscr{L}$), without affecting r_0 (or l_0).
(2) If the rightmost column in $\mathscr{R}$ (or $\mathscr{L}$) is one single coin, move that coin to a new rightmost column of some $\mathscr{J}_{\mu_i}$ (which may be empty initially).	(2) If the rightmost column in some $\mathscr{J}_{\mu_i}$ consists of one coin only, move that coin to a new rightmost column in $\mathscr{R}$ (or $\mathscr{L}$), where $\mathscr{R}$ (or $\mathscr{L}$) is previously non-empty.
(3) Minimum *leftward* coin move in any $\mathscr{J}_{\mu_i}$.	(3) Minimum *rightward* coin move in any $\mathscr{J}_{\mu_i}$.
(4) Let k denote the total number of coins in all of the longest (= lowest) rows from all of the $\mathscr{J}_{\mu_i}$. Remove these k coins, add one more coin to the set, and distribute $k+1$ coins to r_p, $p=0,\ldots,t$ and l_q, $q=0,\ldots,k-t-1$ such that at least all non-zero columns of $\mathscr{R}$ and $\mathscr{L}$ are given coins.	(4) Remove one coin from each column of $\mathscr{R}$ and $\mathscr{L}$. Subtract one coin from this set and distribute the remaining coins on eigenvalue partitions as follows. First, all non-zero columns for all eigenvalues are given one coin each. Remaining coins are assigned to new (rightmost) columns of existing partitions or on new partitions (for new eigenvalues).
Rules 1 and 2 may not make coin moves that affect r_0 (or l_0).	

$\mathscr{B}(A-\lambda B)$ covers $\mathscr{B}(\tilde{A}-\lambda\tilde{B})$:	$\mathscr{B}(A-\lambda B)$ is covered by $\mathscr{B}(\tilde{A}-\lambda\tilde{B})$:
(1) Same as rule 1 above.	(1) Same as rule 1 above.
(2) Same as rule 2 above, except it is allowed only to start a new set corresponding to a new eigenvalue (i.e., no appending to nonempty sets).	(2) Same as rule 2 above, except that J_{μ_i} only consists of one coin.
(3) Same as rule 3 above.	(3) Same as rule 3 above.
(4) Same as rule 4 above, but apply only if there is just one eigenvalue in the KCF or if all eigenvalues have at least two Jordan blocks.	(4) Same as rule 4 above, but apply only if there are no Jordan blocks and coins are used to create one row in the partition for one new eigenvalue, or alternatively, one coin is added to each nonempty column in all existing partitions and possibly to empty columns, but no new partitions are created.
(5) Let any pair of eigenvalues coalesce, i.e., take the union of their sets of coins.	(5) For any eigenvalue J_{μ_i}, divide the set of coins into two partitions so that their union is $\mathscr{J}_{\mu_i}$.

blocks, since this is the operation contrary to removing the only Jordan block of the only eigenvalue in rule 4 for going downwards. Second, the rule may be applied to add coins to all existing columns for all eigenvalues (including already empty columns), since this is the contrary to both removing the largest Jordan block for each eigenvalue, if there are at least two eigenvalues, and to remove the largest Jordan block for a single eigenvalue with more than one Jordan block. Finally, that no new partitions may be created when adding coins to

already existing eigenvalues corresponds to the restriction (in rule 4 for going downwards), that Jordan blocks only can be removed from multiple eigenvalues if all eigenvalues have at least two blocks.

5. STRATIGRAPH SOFTWARE DESIGN ISSUES

5.1. Why Java?

Java is a modern object-oriented language with good support for GUI-implementations. Moreover, it is platform independent, which makes the software portable. When this project initially was started, Java was new and there was also some ideas of making StratiGraph an applet-based tool.

Still, Java does not produce as efficient code as several other programming languages and this has had some influence on the design of StratiGraph. Performance is given priority before memory space, especially in the design of graphics.

5.2. Internal data representation

The goal, when implementing StratiGraph, has been to make it as modular as possible, making it easy to further develop and easy to maintain. The core around which StratiGraph is built is the data representation of the graph. This is a complex dynamic structure with objects representing the nodes and edges and references linking them together. The nodes and edges contain information about the corresponding canonical structure, the position on a virtual grid and references to connected edges.

The structure information has its representation as the partitions or coin tables previously discussed. A coin table is represented as an integer array containing the number of coins in each column. A Jordan structure is associated with a list of coin tables representing the eigenvalue partitions. A Kronecker structure has two extra coin tables representing the right and left singular partitions. The data representation of the structures also contain information about the codimension and whether a structure represents an orbit or a bundle. The graph items also hold some information about the string representation of the structure.

When the graph is expanded, new nodes and edges are calculated and added to the graph structure. If a newly calculated structure already exists within the graph then only a new edge is added otherwise a new node is added.

As mentioned above, each graph item holds its position in a virtual grid. The co-ordinates of the graph items in the virtual grid are not changed when the graph is rescaled or moved in the graphic window, but only when nodes are repositioned or the graph is expanded. The first node generated gets the position (0,0), and new added items get their virtual positions with respect to the (0,0) co-ordinate. StratiGraph uses the virtual grid when it draws the scaled graph on a graphical canvas. A third set of co-ordinates is then used at the view port in StratiGraph. This is needed to handle mouse clicks. In Figure 3, the three co-ordinate systems are illustrated.

5.3. The visualization of the graph

How to present the graph is a non-trivial issue. What should the nodes look like? How should the edges be presented? At first, we wanted the structure information to be presented within the nodes in the graph. But this was only possible for very small graphs where the structures just

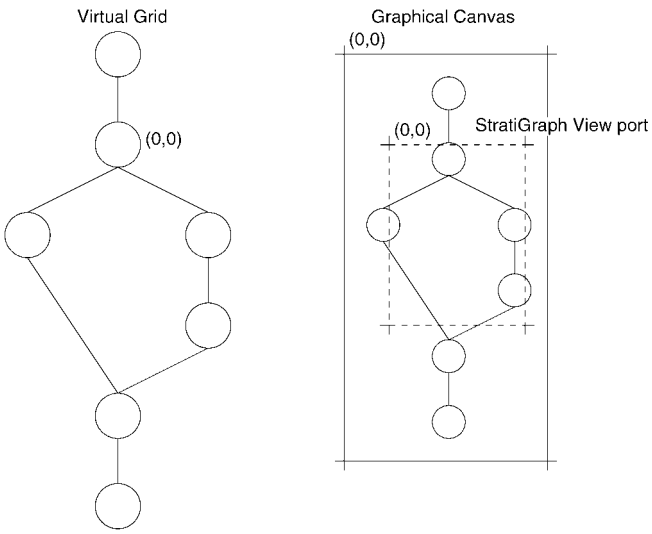


Figure 3. Illustration of the three co-ordinate systems used in StratiGraph.

contained a small number of blocks. Many ideas with popup windows and similar solutions were also eventually dropped, mainly due to limitations in early versions of Java. A good compromise is the field above the graph in the StratiGraph main window, which presents the information about the active node.

Another issue was the look of the edges. Should they be straight or not and should they be allowed to pass over nodes? All these issues ask for a compromise between efficiency and a graph that is easy to follow. However, practical testing showed that a reasonably good node placement algorithm could avoid much of the inconvenience with straight edges. The downside is that they sometimes pass over a node.

5.4. Node placement policy

The previous section briefly elucidated the fact that placement of the nodes in the graph is a vital issue. As a graph is expanded, it is important that it is easy to follow the relation between nodes. The nodes must be placed at a logical position on the basis of the relation to other nodes and to avoid crossings of edges. This is a non-trivial algorithm and several factors are to be considered.

StratiGraph is an interactive tool and the user expects a short response time when expanding a graph. The calculation of new nodes is also a rather time consuming process. This means that it is important that the node placement algorithm is fast and efficient.

Much of the work with StratiGraph has been the development of a fast and yet reasonably good and robust algorithm for the placement of nodes that avoid edge crossings.

The algorithm used in StratiGraph works at one level at the time, and at each level, the following procedure is undertaken:

- On each level, the number of nodes to be placed is known, and so is the number of edges that is not connected to any node on that level but passes it. In order for the graph

not to become too wide, the number of presumptive positions of nodes is limited and based on the number of nodes and passing edges.
- Each node on the level makes a request for a wanted position based on the placement of connected nodes on other levels. Only connected nodes that already have been given a position are considered at this point.
- To avoid that nodes are placed over passing edges, all presumptive positions are given a penalty based on how close they are from the starting point of a passing edge. The closer a node is to such a starting point the more likely it is that an edge will pass over it. The most unsuitable positions is then sorted out.
- Now, the nodes can be sorted by their wanted positions and each node is allotted a position from the remaining most suitable positions.

A more complex algorithm is also implemented where not only penalty for passing edges is added to the presumptive positions, but where each node can give an individual penalty based on the distance to its most wanted position. The placement that gives the least total penalty is then chosen. This gives a better placement of the nodes, but unfortunately, the complexity of the algorithm is also radically increased, which makes it useful only for small graphs.

6. SAMPLE APPLICATIONS

Several characteristics of linear state space models, such as controllability and observability, can be described in terms of matrix pencils (e.g. see References [1, 2, 11]). Let us consider the linear system

$$Ex'(t) = Fx(t) + Gu(t)$$

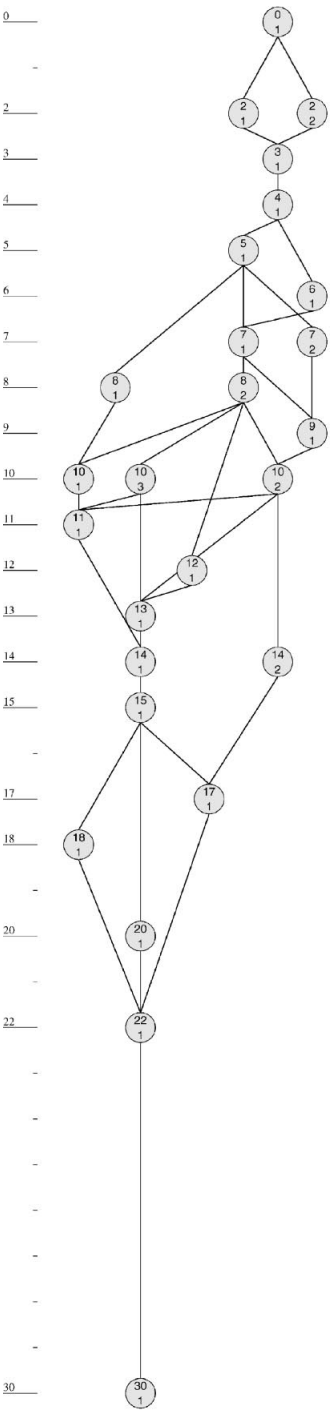
where E and F are $n \times n$ and G is $n \times p$. The system is *controllable* if, starting with $x(0) = x_0$, it is possible to choose the input u to bring the state vector x to an arbitrary state in some finite time t_N. One way to characterize controllability is via the controllability pencil

$$\mathscr{C}(E, F, G) = [G|F] - \lambda[0|E]$$

It has full rank except at $k < n$ values of λ, which correspond to the *uncontrollable modes* of the linear system above.

Controllability decision is an ill-posed problem in the sense that small perturbations in the data may drastically change the behavior of the system. Therefore, it is of great value to know the stratification of the controllability pencil and thereby get qualitative information of nearby canonical structures.

Let us consider a system with three states ($n = 3$) and two inputs ($p = 2$), i.e. we have a 3×5 controllability pencil. In Figure 4, the graph for the bundle stratification of the set of matrix pencils of size 3×5 is displayed. This gives the complete picture, but several of the Kronecker structures represented in the graph are not possible for this application. Let $A \equiv [G|F]$ and $B \equiv [0|E]$, and we assume that E is non-singular. Then it follows that $A - \lambda B$ can only have L_j blocks and finite eigenvalues. The substratification corresponding to these cases are displayed in the top-right part of Figure 4, starting from the node labeled '0 over 1' and ending at the node labeled '14 over 2'. The generic case is $L_1 \oplus L_2$, which corresponds to a controllable system. We remark, that at codimension level 2 there are two structures



Nodes with no L_j^T blocks:

0:1 $L_1 \oplus L_2$
2:1 $L_0 \oplus L_3$
2:2 $2L_1 \oplus J_1(\mu_1)$
3:1 $L_0 \oplus L_2 \oplus J_1(\mu_1)$
4:1 $L_0 \oplus L_1 \oplus J_1(\mu_1) \oplus J_1(\mu_2)$
5:1 $L_0 \oplus L_1 \oplus J_2(\mu_1)$
6:1 $2L_0 \oplus J_1(\mu_1) \oplus J_1(\mu_2) \oplus J_1(\mu_3)$
7:1 $2L_0 \oplus J_2(\mu_1) \oplus J_1(\mu_2)$
7:2 $L_0 \oplus L_1 \oplus 2J_1(\mu_1)$
8:2 $2L_0 \oplus J_3(\mu_1)$
9:1 $2L_0 \oplus 2J_1(\mu_1) \oplus J_1(\mu_2)$
10:2 $2L_0 \oplus J_1(\mu_1) \oplus J_2(\mu_1)$
14:2 $2L_0 \oplus 3J_1(\mu_1)$

Nodes with L_j^T blocks:

8:1 $L_0 \oplus 2L_1 \oplus L_0^T$
10:1 $2L_0 \oplus L_2 \oplus L_0^T$
10:3 $2L_0 \oplus L_1 \oplus L_1^T$
11:1 $2L_0 \oplus L_1 \oplus L_0^T \oplus J_1(\mu_1)$
12:1 $3L_0 \oplus L_2^T$
13:1 $3L_0 \oplus L_1^T \oplus J_1(\mu_1)$
14:1 $3L_0 \oplus L_0^T \oplus J_1(\mu_1) \oplus J_1(\mu_2)$
15:1 $3L_0 \oplus L_0^T \oplus J_2(\mu_1)$
17:1 $3L_0 \oplus L_0^T \oplus 2J_1(\mu_1)$
18:1 $3L_0 \oplus L_1 \oplus 2L_0^T$
20:1 $4L_0 \oplus L_0^T \oplus L_1^T$
22:1 $4L_0 \oplus 2L_0^T \oplus J_1(\mu_1)$
30:1 $5L_0 \oplus 3L_0^T$

Figure 4. Bundle stratification of 3×5 pencils.

$L_0 \oplus L_3$ and $2L_1 \oplus J_1(\mu_1)$. The last case represents a system with one uncontrollable mode, and a controllable subspace of size two. The structure $L_0 \oplus L_3$ represents a system which is controllable, but only for one of the two input variables. The least generic case $2L_0 \oplus 3J_1(\mu_1)$ corresponds to a system with three multiple uncontrollable modes.

Finally, we illustrate a situation where we have specified a Jordan structure $J_6 \oplus J_3 \oplus J_2 \oplus J_1$ and are interested to know the nearby structures above and below in the structure hierarchy. This corresponds to a nilpotent orbit substratification, see Plate 3, where both the blocks and Weyr characteristics of the nearby structures are displayed. Starting from the active node '32:1' and applying the rules for orbits in Table I, we get the structures that are covering the active node by applying a minimum rightward coin move on the partition [432111], resulting in [422211] and [4311111]. By applying a minimum leftward coin move, which can be done in three ways, we get the structures [43221], [441111] and [522111] that are covered by the active node.

7. RELATED STRATIFICATION WORK

StratiGraph builds on the mathematical theory for the stratification of orbits and bundles of matrices and matrix pencils that was recently completed [4]. The mathematical theory for nilpotent matrices goes back to 1961. Assuming well clustered eigenvalues, we may shift all the blocks to be nilpotent. When the eigenvalues are not well clustered, then we consider the bundle case as defined by Arnold [12].

For orbits of matrix pencils, the closure ordering was published by Pokrzywa in 1986 [15], later reformulated by De Hoyos [16] and Elmroth [17]. Recently, Gracia and De Hoyos presented safety neighbourhoods for the necessary conditions in the change of the Jordan and Kronecker canonical structures under small perturbations [18, 19]. A general unifying algebraic theory of degenerations has been obtained for quivers by a number of authors including Abeasis and del Fra [20] and Bongartz [21].

Edelman, Elmroth, and Kågström [4] extend Pokrzywa's results by providing both necessary and sufficient conditions for one Kronecker orbit to cover another. They also present new results for covering and closure relations for the Kronecker bundle case, and propose the use of stratifications in staircase algorithms. A detailed study of the stratification of 2×3 matrix pencils was presented in [22], which also include quantitative results of nearby structures.

The stratification of orbits of pencils associated with problems in control theory has recently been studied by several authors, including Willems [23], Hinrichsen and O'Halloran [24], Boley [25], and Garcia-Planas [26]. For control applications, the pencils of interest typically have no row indices or no column indices.

8. AVAILABILITY AND FUTURE EXTENSIONS

To get a copy of Version 1.1 of StratiGraph, contact one of the authors. The software is distributed as compressed class packages that can be installed and run on any system with a Java 1.1 compatible virtual machine.

Together with the software, some installation advice and the online version of the User's Guide [27] are also available.

The development of StratiGraph is an ongoing project and further functionalities are planned. For example, interaction with Matlab is underway. Matlab routines that calculate distances between different structures in a Jordan or Kronecker stratification are currently under development.

Besides the four types of stratification StratiGraph handles today, extensions for orbits and bundles of pairs, triplets and quadruples of matrices are also planned. All these cases can be seen as substratifications of a general matrix pencil, but are of special interest due to the relationship with various control applications [23, 28, 29, 24].

ACKNOWLEDGEMENTS

We would like to thank Alan Edelman for stimulating discussions regarding stratification tools. Financial support has been received by the Swedish Research Council for Engineering Sciences under contract TFR 222-97-112.

REFERENCES

1. Van Dooren P. The generalized eigenstructure problem in linear system theory. *IEEE Transactions on Automatization and Control* 1981; **AC-26**(1):111–129.
2. Demmel J, Kågström B. Accurate solutions of ill-posed problems in control theory. *SIAM Journal on Matrix Analysis and Applications* 1988; **9**(1):126–145.
3. Doyle P, Bennet RB. *Unlocking the Stratigraphical Record.* Wiley: Chichester, UK, 1998.
4. Edelman A, Elmroth E, Kågström B. A geometric approach to perturbation theory of matrices and matrix pencils. Part II: a stratification-enhanced staircase algorithm. *SIAM Journal on Matrix Analysis and Applications* 1999; **20**(3):667–699.
5. Gantmacher F. *The Theory of Matrices*, *Vol. I and II* (*transl.*). Chelsea: New York, 1959.
6. Van Dooren P. The computation of Kronecker's canonical form of a singular pencil. *Linear Algebra and Its Applications* 1979; **27**:103–141.
7. Demmel J, Edelman A. The dimension of matrices (matrix pencils) with given Jordan (Kronecker) canonical forms. *Linear Algebra and Its Applications* 1995; **230**:61–87.
8. Edelman A, Elmroth E, Kågström B. A geometric approach to perturbation theory of matrices and matrix pencils. Part I: versal deformations. *SIAM Journal on Matrix Analysis and Applications* 1997; **18**(3):653–692.
9. Waterhouse W. The codimension of singular matrix pairs. *Linear Algebra and Its Applications* 1984; **57**:227–245.
10. Demmel J, Kågström B. The Generalized Schur decomposition of an arbitrary pencil $A-\lambda B$: robust software with error bounds and applications. Part I: theory and algorithms. *ACM Transactions on Mathematical Software* 1993; **19**(2):160–174.
11. Demmel J, Kågström B. The generalized Schur decomposition of an arbitrary pencil $A-\lambda B$: robust software with error bounds and applications. Part II: software and applications. *ACM Transactions on Mathematical Software* 1993; **19**(2):175–201.
12. Arnold VI. On matrices depending on parameters. *Russian Mathematical Surveys* 1971; **26**:29–43.
13. Kågström B. RGSVD—an algorithm for computing the Kronecker canonical form and reducing subspaces of singular matrix pencils $A-\lambda B$. *SIAM Journal on Statistical and Scientific Computing* 1986; **7**(1):185–211.
14. Stanley RP. *Enumerative Combinatorics*. Wadsworth & Brooks/Cole: 1986.
15. Pokrzywa A. On perturbations and the equivalence orbit of a matrix pencil. *Linear Algebra and Its Applications* 1986; **82**:99–121.
16. De Hoyos I. Points of continuity of the Kronecker canonical form. *SIAM Journal on Matrix Analysis and Applications* 1990; **11**(2):278–300.
17. Elmroth E. On the stratification of the Kronecker canonical form. *Report UMINF-95.14*, Department of Computing Science, Umeå University, S-901 87 Umeå, Sweden, May, 1995.
18. Gracia JM, De Hoyos I, Velasco FE. Safety neighbourhoods for the invariants of the matrix similarity. *Linear and Multilinear Algebra*, 1996, to appear.
19. Gracia JM, De Hoyos I. Safety neighbourhoods for the Kronecker canonical form. *SIAM Journal on Matrix Analysis and Applications*, 1999, submitted.
20. Abeasis S, Del Fra A. Degenerations for the representations of a quiver of type $\mathscr{A}_m$. *Journal of Algebra* 1985; **93**:376–412.

21. Bongartz K. On degenerations and extensions of finite dimensional modules. *Advances in Mathematics* 1996; **121**(2):245–287.
22. Elmroth E, Kågström B. The set of 2-by-3 matrix pencils—Kronecker structures and their transitions under perturbations. *SIAM Journal on Matrix Analysis and Applications* 1996; **17**(1):1–34.
23. Willems JC. Topological classification and structural stability of linear systems. *Journal of Differential Equations* 1980; **35**:306–318.
24. Hinrichsen D, O'Halloran J. Orbit closures of singular matrix pencils. *Journal of Pure and Applied Algebra* 1992; **81**:117–137.
25. Boley D. The algebraic structure of pencils and block toeplitz matrices. *Linear Algebra and Its Applications* 1998; **279**:255–279.
26. Garcia-Planas MI. Kronecker stratification of the space of quadruples of matrices. *SIAM Journal on Matrix Analysis and Applications* 1998; **19**(4):872–885.
27. Johansson P. StratiGraph user's guide. Version 1.1. *Report UMINF-99.11*, Department of Computing Science, Umeå University, SE-901 87 Umeå, Sweden, April 2000.
28. Garcia-Planas MI, Margret MD. An alternative system of structural invariants of quadruples of matrices. *Linear Algebra and Its Applications* 1999; **291**:83–102.
29. Gracia JM, De Hoyos I, Zaballa Z. Perturbation of linear control systems. *Linear Algebra and Its Applications* 1989; **121**:353–383.

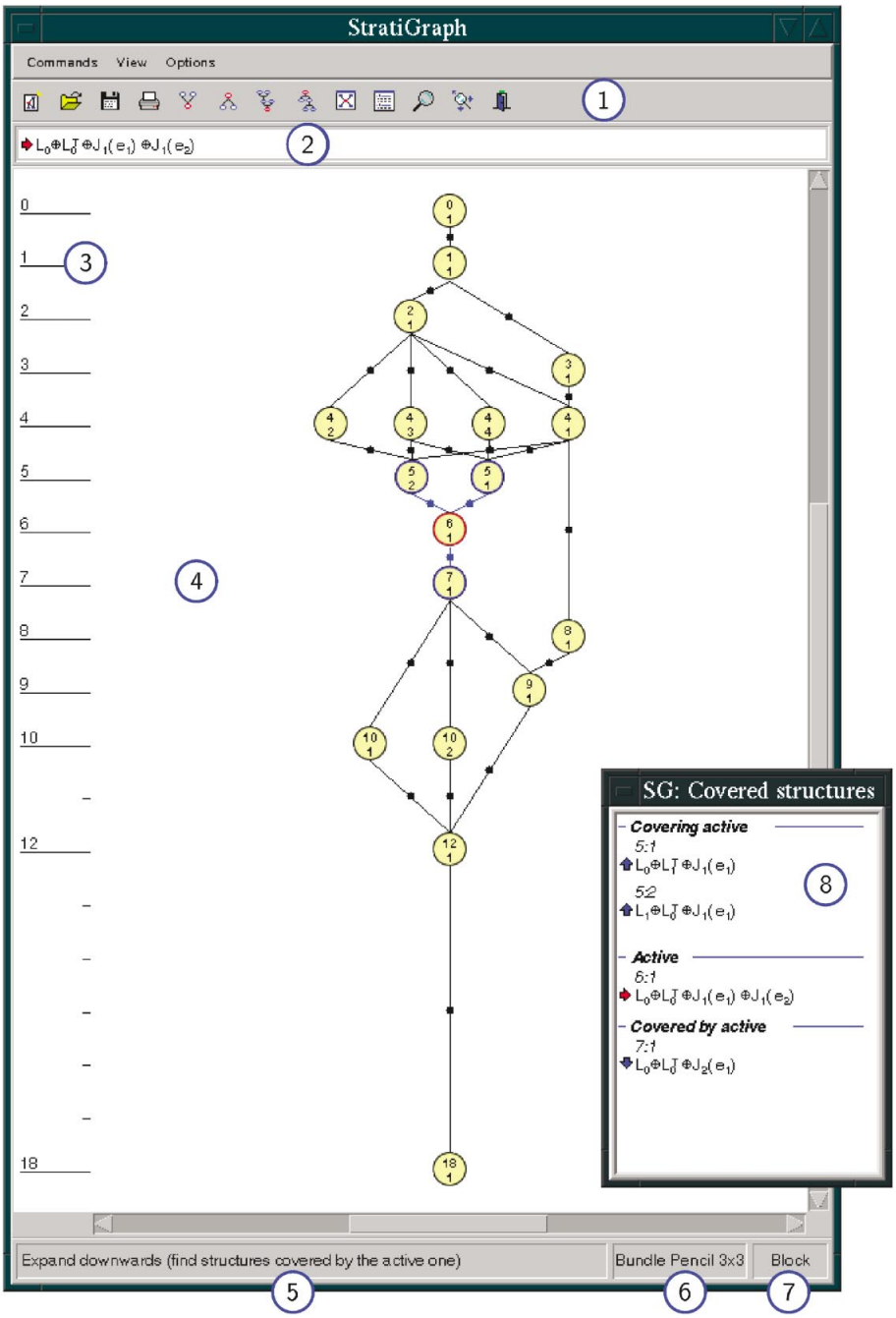


Plate 1. StratiGraph's main window.

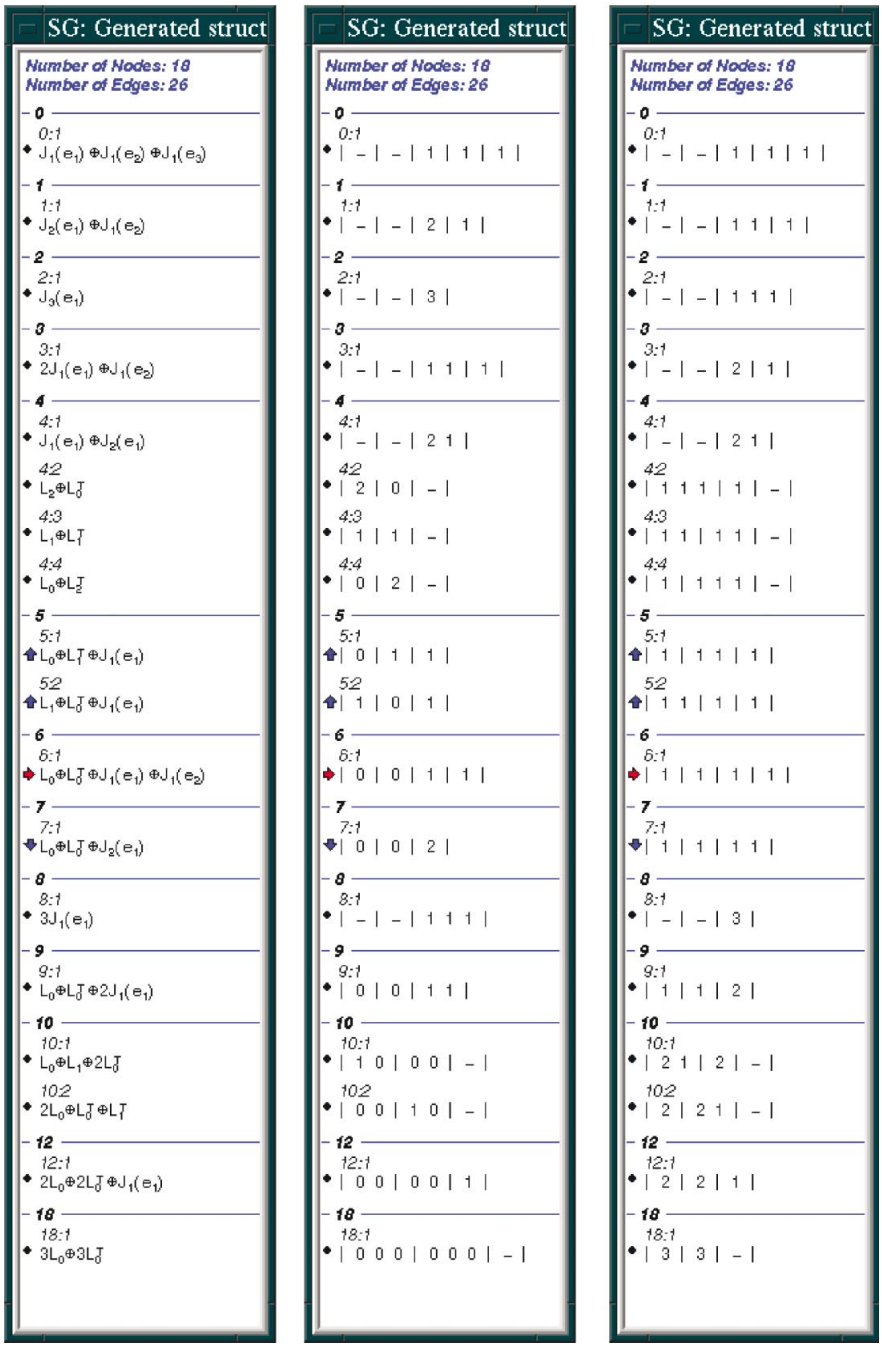


Plate 2. The ‘Generated structures’ window shows the canonical structure information in block, Segre or Weyr notation (here we illustrate all three). The Segre and Weyr notations for different types of blocks are separated by the ‘|’ symbol and given in order for right singular blocks, left singular blocks, and for Jordan blocks of the first eigenvalue, second eigenvalue, etc.

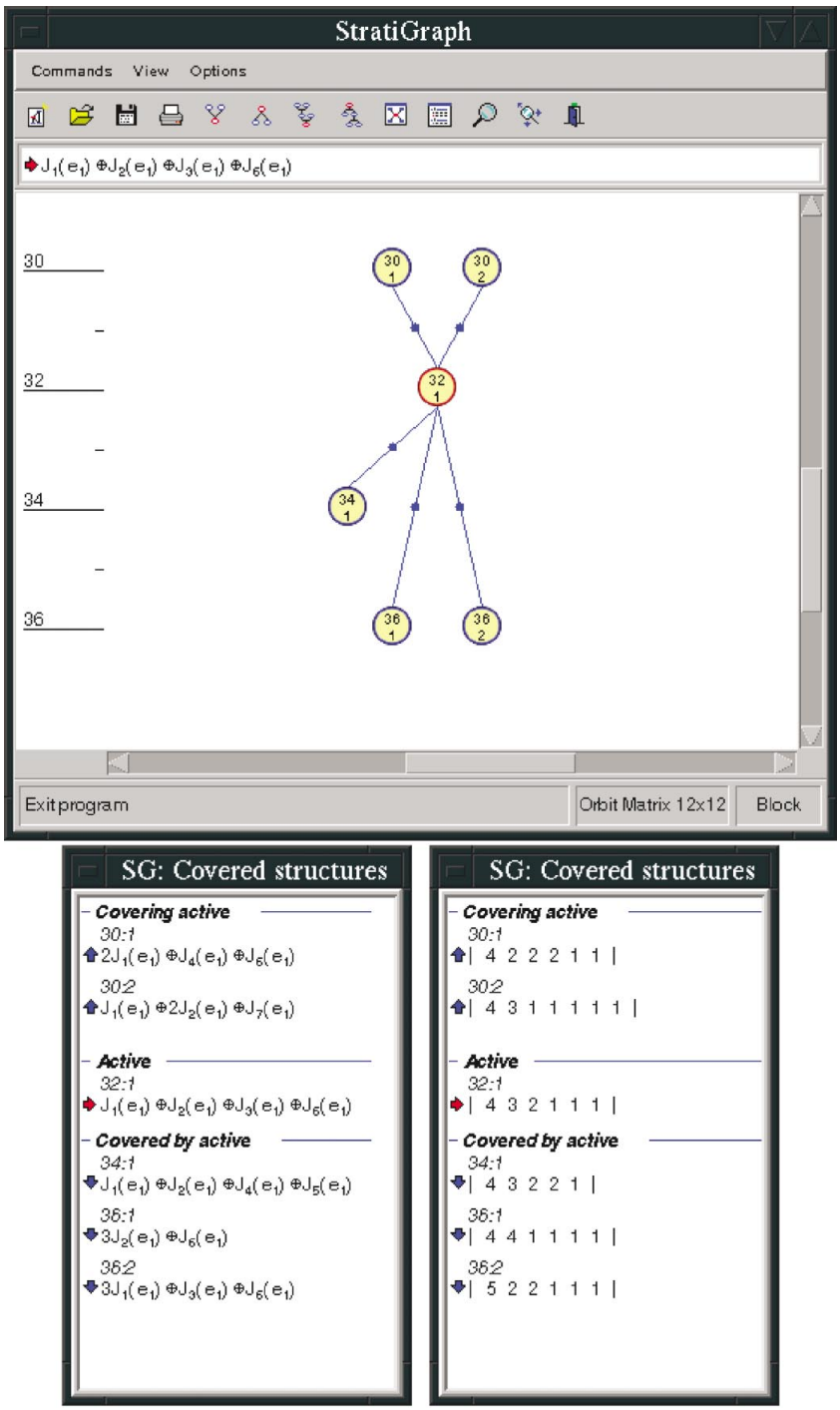


Plate 3. Nearest neighbours of $J_6 \oplus J_3 \oplus J_2 \oplus J_1$ (orbit case).