N90-22988

An Assessment of Multibody Simulation Tools for Articulated Spacecraft

Guy K. Man and Samuel W. Sirlin
The Jet Propulsion Laboratory
The California Institute of Technology
4800 Oak Grove Drive, Pasadena, CA 91109

Introduction

A survey of multibody simulation codes was conducted in the spring of 1988, to obtain an assessment of the state of the art in multibody simulation codes from the users of the codes. This information will be used to evaluate the need to develop future codes. If this need is established, it will also be used to guide the development of future capabilities. A questionnaire, covering 30 issues, was developed for this purpose. Sixty questionnaires were sent out to 50 organizations. We received 40 replies from 30 organizations covering 21 simulation codes.

This survey covers the most often used articulated multibody simulation codes in the spacecraft and robotics community. There was no attempt to perform a complete survey of all available multibody codes in all disciplines. Furthermore, this is not an exhaustive evaluation of even robotics and spacecraft multibody simulation codes, as the survey was designed to capture feedback on issues most important to the users of simulation codes. We must keep in mind that the information received was limited and the technical background of the respondent varied greatly. Therefore, this paper only reports the most often cited observations from the questionnaire. In this survey, it was found that no one code had both many users (reports) and no limitations.

The paper is organized as follows. The next section is a report on multibody code applications. Following applications is a discussion of execution time, which is the most troublesome issue for flexible multibody codes. The representation of component flexible bodies, which affects both simulation setup time as well as execution time, is presented next. Following component data preparation, two sections address the accessibility or usability of a code, evaluated by considering its user interface design and examining the overall simulation integrated environment. A summary of user efforts at code verification is reported, before a tabular summary of the questionnaire responses. Finally, some conclusions are drawn.

Applications

It is not surprising that general purpose multibody simulation codes are used to address spacecraft and robotics with articulated elements, because of the complexity of the equations

Doug Petesch suggested that users explore supercomputing capabilities before presuming that the solution required parallel processing. He also mentioned the advantage that comes from doing software development as well as production runs on the supercomputer. The value of a hypothetical teraflop machine was discussed. It was agreed that while valuable, it would not solve all the problems—such as numerical difficulties with higher order systems. In addition some thought that the same speed could be achieved at a much lower cost using parallel processing.

Dick Vandervoort of DYNAC referred back to John Doyle's comments about uncertainty, noting that multibody dynamics modeling gives highly detailed models—more detailed than the control designer needs or wants. In fact most of it is uncertainty, and what we need is a way to model that uncertainty in a way that can be used in the control design

Perhaps John Hedgepeth's comment best summed up the reason to move forward with Computational Control. Summarizing the comments of several speakers, he noted that we are proceeding ahead conservatively in control system design for planned missions, and that without improvements in technology—we are choosing to design only spacecraft that we already know how to design.

of motion. Some codes are equipped to simulate ground vehicles or aircraft. The actual applications include control design and verification, deployment simulation, machine design, impact simulation, tether simulation, explosion simulation, and human in-the-loop real time simulators for the Space Shuttle and Space Station. About 70% of the users reported that multibody codes are used for control system design and verification. The remaining 30% apply multibody codes for system design and dynamics studies. It is also reported that multibody codes are used to check other codes. For example, a rigid body code might be used to check a more complex flexible multibody code.

System order exercised varies from a robot arm with less than 10 states to a Space Station model with 150 states. One user reported trying a model with over two hundred states with modes up to 1000 Hz using DISCOS. Flexible body systems usually are higher order systems than rigid body systems, and the simulation duration is usually shorter. Simulation duration does however vary according to the application.

Execution Time

Excessive execution time is one of the two most cited shortcomings of current multibody codes. This concern applies especially to flexible multibody simulations. Existing codes are so slow for most problems that it is impossible to use them during the design phase when quick turnaround is needed. Highly simplified flexible models or even rigid body models must suffice, even though the control system designer fully recognizes that the result may be inaccurate. For example, consider a 20 degree of freedom flexible system with flexible modes less than 100 Hz. The CPU time to real time ratio for this example is 200/1 on a VAX class computer. As the system order becomes higher, the computational load will become much more stressing because the computational load scales as N^3, for the current codes, where N is the number of degrees of freedom.

It is interesting to note from the survey that the simulation durations for flexible body problems are usually short (seconds). Very few users are using flexible codes for long runs. From the data, simulation duration is seen to be inversely proportional to the system order. This telltale observation indicates that flexible multibody simulation codes are not being used extensively, because long duration simulation costs are too high. One of the respondents summarizes this quite well - "Multibody run times are seldom acceptable. You either pay the price or get approximate answers."

Getting an approximate answer is the only way to reduce excessive execution time for given hardware and software. Approximation of flexible multibody systems is done most commonly by reducing the order of the individual flexible body dynamics, in fact often reducing the bodies down to rigid bodies. More than one user reported that fast rigid body simulations were used to check the more complex flexible body simulations. If appropriate, the more complex flexible body simulation is abandoned in favor of the faster rigid body code. The model reduction approach has not been included in this survey. But some observations of this method are discussed in the next section.

Component Model Representation

One of the major concerns in modeling a flexible multibody system is how to obtain data for the component elastic bodies. Among the flexible multibody codes surveyed, all except one code use the modal representation for flexible bodies. The primary benefit of using a modal representation is "simplicity," which can lead to reduced computation time. There are a variety of modes one can use to represent the individual bodies of a multibody system, such as Hurty modes (Craig Bampton modes) or cantilever modes. The modes of each body have to be supplied to the multibody code which will then synthesize the modes of the component bodies to arrive at a system model for time simulation.

The body of knowledge of choosing the best component mode set for system synthesis is called component mode synthesis. This includes both choosing the type of modes to use (possibly several) as well picking out which of the modes must be retained for simulation fidelity. The various component mode sets were developed for this purpose, however they were developed for systems with no articulating components, essentially static system geometry. Flexible multibody codes, on the other hand, were developed to model dynamical systems with varying geometry. The applicability of component mode synthesis results to multibody analysis is not well understood and care must be taken in each specific case to ensure that the results are correct. One question that needs to be asked often is whether or not a set of component body modes is complete over the range of interest of the articulation angles (or translations). After the system is synthesized in the multibody code, a check can be made against a higher fidelity linear model such as a NASTRAN system model only for a fixed geometry.

It was pointed out in the previous section that model reduction is one of the commonly used methods to obtain an approximate solution. This is a necessary step for obtaining a reasonable simulation computational time. A variety of methods exist to pick out the significant modes. These arose out of the the slightly different model reduction problems of control theory. What model reduction criterion should one use to obtain an adequate approximate multibody model? No definitive answer exists, but the answer certainly depends on how the simulation is to be used. For example, if the simulation is to be used for control system design, then controllability and observability criteria may play a part. It may also be useful to consider the control design problem simultaneously with the model reduction problem. In some multibody codes the level of modal approximation used (for example the DISCOS mass distribution options) ties directly into the flexibility data preparation process, compounding the data preparation problem.

Considering these issues, it is not surprising that the survey indicates flexible data requirements are obscure, challenging, poorly treated and inadequate. Considerable research remains to be done in component model representation.

User Interface

One of the most common complaints concerning current multibody simulation codes is the lack of interactive model setup capability. Codes that have a friendly input format win praise from the users. Interactive, menu driven type preprocessors with good error messages are

indicated as highly desirable. Incomplete documentation of the code was the second most cited complaint. Sophisticated users also desire well commented source code, especially if the user manual is inadequate. Online manuals and documentation were mentioned by several users as highly desirable. For flexible multibody codes, the availability of the theoretical background on which the code is based is very important. In addition to documentation, application examples are found to be an effective way to train users. It was also indicated that examples are needed for modal data preparation.

At the output interface, data retrieval flexibility and graphics seemed to be most important. Users were divided on whether they wanted built-in plotting capabilities or whether they simply wanted the ability to export data to their own favorite plotting programs. The survey did not indicate any major concerns in this area.

Integrated Environment

Multibody codes are sometimes used as stand alone tools but more recently complex flexible multibody codes are often used in conjunction with other software such as finite element, control analysis, or optics codes. In the latter environment, the multibody code becomes a fundamental building block of an integrated design and analysis environment. Some of the codes surveyed, such as TREETOPS, are actually a microcosm of such an environment. This is a natural development trend, because multibody systems are becoming more complex, forcing system engineering work to become multidisciplinary. The survey pointed out that no environment can satisfy everybody. Every organization has its own culture and emphasis. Everybody has some pet code. It is therefore perhaps wise to treat multibody codes as modular building blocks which can be easily integrated into a bigger environment. If this is the approach we adopt, then there is a need to define a standard interface for data passage.

There are a number of additional capabilities that users found helpful beyond the generation and integration of equations of motion. These are: a library of joints sensors and actuators, the ability to obtain constraint forces, transfer functions, Jacobians, and key state matrices for external control analysis, the ability to incorporate user defined subroutines, and of course a model reduction preprocessor.

Verification

A number of codes have been independently checked by users with other codes. Most users utilize simple test cases and the principles of mechanics to check simulation results. Of the test cases reported, none are designed to check flexible body effects. This is not surprising, because if component model data generation is not well understood, the verification of the model must also be nontrivial. More work should be done in this area so as to add confidence in the simulation results and the flexible multibody codes.

It is suggested that a set of standard test cases be collected, which can be used by the community to check both existing and new simulations codes. The set should include simple test cases with known simulation results as well as actual experiments with test data. This

collection process has been started by NASA and the University of Iowa.

Tabulated Data

The user questionnaire data has been summarized in the following tables. Table 1 gives the availability of the multibody code, as well as how many respondents used each code and how extensive the information provided by the users was. Almost all of the software is available, either commercially or in the public domain (COSMIC).

Table 2 gives various details of how users used the codes, and what code features they liked or felt needed improvement. Typical applications and whether or not the run times were acceptably fast are given, as well as what component data the code required and, in general, how easy the code was to use. User comments are included on the quality of the documentation as well as what features of the code they liked, and what features could be added to improve productivity. Following are some comments on the individual columns.

There was a wide range in documentation quality, from "clear and precise" to "inadequate overall," which in general indicates how much care the developer took with making the code easy to use.

There was very little overlap from code to code of features that the users appreciated. Some users liked the user interface of some codes, while one code was notable for animation, and another for being database driven. Two features that did show up more than twice were the ability to add user-defined subroutines, and a library of application modules.

There was also a wide variety of additional features that the users desired, again with surprisingly few common needs from code to code. Significant needs included better documentation (mentioned for four codes), and a better data interface capability (also mentioned for four different codes).

The acceptability of the simulation run times was much more uniform. The rigid body codes were all considered acceptably fast, while the flexible body codes were, in general, acceptable only for small problems.

There was considerable variety in the user comments on the type of data needed for the representation of flexible components, indicating the lack of maturity of this area.

It is important to keep in mind that some multibody codes have been around for a long time and so have an extensive user and applications base, while other codes are new and have only been used by relatively few people on a small number of examples.

Conclusions

Examining the reports of existing multibody code users, three facts become evident:

1. It is difficult to use flexible multibody codes in design and analysis due to the long execution times for realistic problems,

2. Representation of component flexible bodies is poorly understood, and

3. Not enough thought has been given to the user interface by the code developers.

At the Workshop on Multibody Simulation in 1988, JPL made a commitment to coordinate a multibody code verification, as part of an ongoing effort of the whole multibody simulation community. This survey report, as well as the verification library begun jointly with the University of Iowa, represent the first steps toward meeting that commitment.

Acknowledgement

The Research described in this paper was performed by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

TABLE 1 MULTIBODY SIMULATION CODES INCLUDED IN THE SURVEY

Code	Contact	# of Replys	Information Received	Availability	Host Computer
ADAMS	Rajiv Rampali MDI Inc. 3055 Plymouth Rd. Ann Arbor, MI 48105	3	extensive	yes, commercial	• Cray • Prime • NEC • Cyber • IBM • SGI • Alliant • HP • Convex • Sun • DEC • Apollo • Intergraph
ALLFLEX	Jimmy Ho Lockheed Missiles & Space Co. Space System Division 1111 Lockheed Way Sunnyvale, CA	1	moderate	yes	• CRAY • VAX
AUTOLEV	David Levinson 93-30/250 Lockheed Palo Alto Research Lab 3251 Hanover Street Palo Alto, CA 94304	1	moderate	yes, commercial	IBM PC/CLONE
CONTOPS	John Sharkey NASA MSFC Code ED 12 Marshall Space Flight Center Alabama 35812	4	extensive	yes, COSMIC	• VAX 11/750 • MICRO VAX • HP 9000
DADS	CADSI, Inc. P.O. Box 203 Oakdale, IA 52319	5	extensive	yes, commercial	• VAX • Multiflow • Alliant • Prime • Apollo • Silicon Graphics • Cray • IBM PC • DEC • Computervision • IBM • Cyber • Sun • HP

Code	Contact	# of Replys	Information Received	Availability	Host Computer
DAMS	A.A. Shabana P.O. Box 4348 Dept. of ME U of Illinois @ Chicago Chicago, IL 60680	1	limited	yes	IBM
DISCOS	COSMIC U. of Georgia Computer Service Annex 382 E. Broad Street Athens, GA 30602	7	extensive	yes, COSMIC	• IBM 3090, 770 • VAX 11/780, 750 • MICRO VAX
DYNOCOMBS	Ron Huston U of Cincinatti Dept. of ME & IE Cincinatti, OH 45221-0072	1	moderate	yes	• IBM 4043 • VAX 1170
DYNA/EF3	Jim Lawrence NASA JSC Mail Code EF3 Houston, TX 77058	1	moderate	yes	Gould Concept 32
DYNAMUS	Farid Amirouche Dept. of ME P.O. Box 4348 U of Illinois @ Chicago Chicago, IL 60680	1	extensive	yes	• IBM 370 • VAX 11/780

CODE	DOCUMENTATION	WELCOME FEATURES	ADDITIONAL FEATURES DESIRED	USER EXPERIENCE	TYPICAL APPLICATIONS	EASY TO USE	RUN TIME ACCEPTABILITY	FLEXIBLE COMPONENT DATA
Generic Manipulator Simulation	• Moderately documented • Source available and commented	• Open and closed chain dynamics and loads	• Better I/O • Flexbody dynamics	limited	• Manipulator	yes	yes	Does not model flexibility
LATDYN	• Manual available • Source available and commented	• Easy to use finite element code • Modeling of control forces • Macros	• Need restart • Better naming convention • Available for newer computers • Static Analysis • Interactive preprocessor	moderate	• RMS slew on spacestation • Impact, lockup	yes	Problem dependent	Easy, finite element approach
MIRRORS	Partial, subroutines are not completely documented	• Database driven	• More generic N body code	moderate	• Loads • Shuttle RMS	no	• Not acceptable for flexible problems	• Lumped mass • Craig Bampton
MULTFLEX	• User manual • Source not available, lightly documented	• Easy to change configuration • Symbolic pre-processor • Code generation capability	• Better documentation • Graphic preprocessor • Torsional flexibility • Transfer function	moderate	• Shuttle RMS • Spacecraft	• Easy setup	• Problem dependent • Acceptable on Cray	• Mode shape functions

Code	Contact	# of Replys	Information Received	Availability	Host Computer
MULTFLEX	Martin Tong The Aerospace Corp. MS (M4/972) P.O. Box 92957 Los Angeles, CA 90009-2957	4	moderate	proprietary	• CDC Cyber 175 • Cray X-MP
NBOD2	COSMIC University of Georgia Computer Services Annex 382 E. Broad Street Athens, GA 30602	1	limited	yes, COSMIC	VAX 11/785
SD/EXACT	Dan Rosenthal 11585 Silvergate Drive Dublin, CA 94568	6	extensive	yes, commercial	• VAX 11/785 • IBM 3090 • VAX Station II • Sun Workstation • HP-9000 • MICRO VAX
SPASIS	COSMIC University of Georgia Computer Service Annex 382 E. Broad Street Athens, GA 30602	1	moderate	yes, COSMIC	• VAX 8650 • Harris-800
Station Control Simulation	John W. Sunkel NASA JSC Mail Code EH2 Houston, TX 77058	1	limited	yes	CYBER 830
TREETOPS	John Sharkey NASA MSFC Code ED 12 Marshall Space Flight Center Alabama 35812	3	moderate	yes, COSMIC	VAX

TABLE 2 USER EXPERIENCE

CODE	DOCUMENTATION	WELCOME FEATURES	ADDITIONAL FEATURES DESIRED	USER EXPERIENCE	TYPICAL APPLICATIONS	EASY TO USE?	RUN TIME ACCEPTABILITY	FLEXIBLE COMPONENT DATA
ADAMS	• Well documented • User's & Application Manual • Theory Notes • Source not available	• 9 CAD/CAE Interface • Linearization, Modal Analysis • Large Joint Library • Wide Variety of Forces • Accurate Flexibility • Full Feature Restart • Interface to MatrixX • Postprocessor with Animation • User Subroutines	• Built in control elements • Theory manual	•Extensive (>100 Engineers trained at MDI) •Fully staffed hotline support	AUTOMOTIVE • Subsystem & system modeling • Suspension Design • Ride Analysis AEROSPACE • Robotics • Spacecraft Simulation • Mechanism analysis and design	• Yes • 9 commercial preprocessors available • 3 high-end commercial graphics animation post-processors	Problem Dependent	• ANSYS/NASTRAN reduced stiffness Matrix • Direct physical properties • Handles all geometric nonlinearities • Standard beam and field elements
ALLFLEX	• Documentation • Source not available	• Easy and fast to assemble simulation	• Menu driven input	extensive	Spacecraft	yes	Problem dependent	Format is well defined
AUTOLEV	• Documented • On-line help command • Source not available	• Very versatile • PC based • Explicit symbolic equations	None thus far	moderate	• Robotic device • Spacecraft	yes	yes	Handles systems of rigid bodies only
CONTOPS	• I/O well documented • Inner workings not well documented • Code comments sketchy	• Interactive preprocessor • Plotting nicely done	• More help in error tracing • Modal data prep examples • More application examples • Sensor model • More boundary conditions for flex data • Frequency response data • Expand output list	extensive	• Robots • RMS • Spacecraft • Robot experiment • Smart Structures	yes	Fair, not acceptable for flexible problems	• More flexibility in flex-body representation • Flexible augmentation is obscure
DADS	• Well documented • Source not available • User's manual • Examples manual	• CAD/CAE interfaces • Preprocessor • Control system modeling elements • Postprocessor • Extremely stable integration	• More sophisticated postprocessor • Interface to control analysis package • Stop/restart • Error messages • Control Analysis capability	• Extensive • Fully staffed hotline support	• Servo control of robot arms • Vehicle simulation • Mechanisms • Spacecraft	yes	Problem dependent	• Component modal synthesis • Interfaces to NASTRAN, ANSYS, ABAQUS • Handles all geometric nonlinearities
DAMS	Source is available but not well commented	User subroutine facility	not available	moderate	• Machine design • Robot • Impact	Mixed	Problem Dependent	Normal modes

CODE	DOCUMENTATION	WELCOME FEATURES	ADDITIONAL FEATURES DESIRED	USER EXPERIENCE	TYPICAL APPLICATIONS	EASY TO USE	RUN TIME ACCEPTABILITY	FLEXIBLE COMPONENT DATA
DISCOS	• Source code commented • Documentation inadequate overall	• Very general • User facility subroutine • Documentation of theory (but not complete) • Examples • Very few bugs now • Wide user community • Trustworthy	• Much better documentation • Much better user interface • Faster code • Better numerical stability • Automatic redimensioning	extensive	• Spacecraft • Rapid retargeting maneuvers • Deployment • Propellent slosh • Check other code	• No • Need thorough understanding of code	• Problem dependent • Too slow for high order systems	• Normal modes • Any mode shape
DYNOCOMBS	• Well documented • Source available	• Efficiency • Broad Applicability	• Animated output • Menu driven input	moderate	• Towing of submerged cables • Robotic • Human body modeling	yes	yes	Flexibility is modeled at joints
DYNA/EF3	• Mathematical model documentation • Software design documentation • Source available and commented	• Extensive vehicle library	• Expert systems shell • Better control of data time, transport delay	moderate	• Human-in-the-loop real time simulator for shuttle and space station • Shuttle RMS shuttle/station docking	fair	yes (real time)	Does not model flexibility
DYNAMUS	Yes, clear, precise	• Friendly input format • Automatic elimination of singularities due to constraints • Handles geometric nonlinearities	N/A	moderate	Constraint Tree, Flexible Spacecraft deployment, Manipulator	yes	good	Normal modes Component mode analysis
FB2	• Not well documented • Source is commented	• Fast • Lots of options	• Better I/O • Better documentation • Large rotation between bodies	limited	• Spacecraft • Robot	yes	yes	Craig-Bampton modes

CODE	DOCUMENTATION	WELCOME FEATURES	ADDITIONAL FEATURES DESIRED	USER EXPERIENCE	TYPICAL APPLICATIONS	EASY TO USE	RUN TIME ACCEPTABILITY	FLEXIBLE COMPONENT DATA
FLEXIM	Well documented	• Very few bugs	Not available	limited	• Spacecraft	Yes, but initial investment to learn the code is high	N/A	• Hurty modes • A special NASTRAN postprocessor exists
Generic Manipulator Simulation	• Moderately documented • Source available and commented	• Open and closed chain dynamics and loads	• Better I/O • Flexbody dynamics	limited	• Manipulator	yes	yes	Does not model flexibility
LATDYN	• Manual available • Source available and commented	• Easy to use finite element code • Modeling of control forces • Macros	• Need restart • Better naming convention • Available for newer computers • Static Analysis • Interactive preprocessor	moderate	• RMS slew on space station • Impact, lockup	yes	Problem dependent	Easy, finite element approach
MIRRORS	Partial, subroutines are not completely documented	• Database driven	• More generic N body code	moderate	• Loads • Shuttle RMS	no	• Not acceptable for flexible problems	• Lumped mass • Craig Bampton
MULTFLEX	• User manual • Source not available, lightly documented	• Easy to change configuration • Symbolic pre-processor • Code generation capability	• Better documentation • Graphic preprocessor • Torsional flexibility • Transfer function	moderate	• Shuttle RMS • Spacecraft	• Easy setup	• Problem dependent • Acceptable on Cray	• Mode shape functions

SPATIAL OPERATOR ALGEBRA FRAMEWORK FOR MULTIBODY SYSTEM DYNAMICS

G. Rodriguez, A. Jain, and K. Kreutz
Jet Propulsion Laboratory/California Institute of Technology
4800 Oak Grove Drive, MS 198-219, Pasadena, CA 91109

Abstract: This paper describes the Spatial Operator Algebra framework for the dynamics of general multibody systems. The use of a spatial operator–based methodology permits the formulation of the dynamical equations of motion of multibody systems in a concise and systematic way. The dynamical equations of progressively more complex rigid multibody systems are developed in an evolutionary manner beginning with a serial chain system, followed by a tree topology system and finally, systems with arbitrary closed loops. Operator factorizations and identities are used to develop novel recursive algorithms for the forward dynamics of systems with closed loops. Extensions required to deal with flexible elements are also discussed.

1 Introduction

The field of multibody dynamics is currently being challenged in two major ways. The increase in the size and complexity of spacecraft systems requires the development of tools that not only help manage the complexity of such systems, but also facilitate the development of novel dynamics formulation techniques and solution algorithms. Areas such as robotics involve multibody systems consisting of multiple robot manipulators interacting with each other and with complex environments. These are multibody systems with not only constantly time–varying topological structure, but also ones in which the constituent bodies change with time. Coping with this aspect requires versatile and flexible dynamics simulation tools.

In this paper, the Spatial Operator Algebra Framework [1] is used to develop a systematic procedure for concisely formulating the equations of motion and derive spatially recursive forward dynamics algorithms for multibody systems. The equations of motion of progressively more complex rigid multibody systems such as serial chains, tree topology systems and finally closed chain systems are developed. Operator factorizations and identities are then used to obtain efficient spatially recursive algorithms for the forward dynamics of such systems. Extensions to handle flexible link elements are also discussed.

2 Equations of Motion

We begin by briefly describing the coordinate-free *spatial notation* used throughout this paper. Given the linear and angular velocities v and ω, the linear force F, and moment N at a point on a body, the *spatial velocity* V, *spatial acceleration* α and the *spatial force* f in $\mathcal{R}^6$ are defined as follows:

$$V \triangleq \begin{pmatrix} \omega \\ v \end{pmatrix}, \quad \alpha \triangleq \dot{V}, \quad f \triangleq \begin{pmatrix} N \\ F \end{pmatrix}$$

The *rigid body transformation operator* $\phi(.) \in \mathcal{R}^{6\times 6}$ is defined as:

$$\phi(l) \triangleq \begin{pmatrix} I & \tilde{l} \\ 0 & I \end{pmatrix}$$