The feasibility of using conventional proportional-integral-derivative (PID) control and an alternative optimal control to perform the pointing and tracking functions of the Space Station solar dynamic power module is investigated. A very large state model of 6 rigid body modes and 272 flexible modes is used in conjunction with classical linear-quadratic-Gaussian (LQG) optimal control to produce a full-order controller which satisfies the requirements. The results are compared with a classically designed PID controller that was implemented for a much smaller (6 rigid body, 40 flexible modes) model. From the results, the feasibility of a conventional PID solar dynamic control solution with a reduced-order model, which satisfies the basic system pointing and stability requirements, is suggested. However, the conventional control design approach is shown to be very much influenced by the order reduction of the plant model, i.e., the number of retained elastic modes from the full-order model. This eventually suggests that for a complex large space structure, such as the Space Station Freedom solar dynamic module, application of conventional control system design methods may not be adequate. Instead we are suggesting the design of an optimal control algorithm.

A major difficulty with designing LQG controllers for large models is solving the Riccati equation that arises from the optimal formulation. The appendix describes an algorithm that can be used to solve large matrix Riccati equations. Here we have incorporated a previously-reported Riccati solver that is based on a Padé approximation to the matrix sign function. We also derive a symmetric version of this algorithm for the special class of Hamiltonian matrices thereby yielding, for large problems, a near two-fold speed increase over a previous algorithm. A typical computer run for this problem takes about six minutes, thus making large-order models a possibility for initial "paper design" studies. In this way, applicability of optimal control design for large space structures becomes practical.
Computational and Design Issues

Designing effective controllers for large subsystems of the proposed Space Station gives rise to several computational as well as controller design issues. The lightweight truss structure of the station allows for a large number of lightly damped flexible modes and implies that movement of large subsystems such as the solar dynamic (SD) power modules will cause vibration and interaction throughout the entire station. Therefore, apart from the tracking specification of the power module, vibration suppression is of paramount importance. A first step in a controller design for such a system could be to use a very large-order plant that has been developed in a finite element code such as NASTRAN and design "on paper" a full-order controller for this model. A second step could be to reduce the order of the model using standard model reduction schemes and then redesign a lower-order controller to operate on this plant. At first glance, it seems that designing a controller for the unreduced model is fruitless because a reduced-order controller must be implemented to be of practical value (the full-order controller is computationally too expensive). However, the full-order controller on the unreduced plant will give benchmark performance that the reduced-order controller can try to meet. In practice, any controller for a system which contains an infinite number of modes must include some gain rolloff so that the unmodeled higher-frequency modes will not be excited. We will not pursue this point because the major thrust of this paper is to illustrate the feasibility of design with high-order plants and to compare the simulated performance of a controller on a high-order plant against the performance of a lower-order controller on the reduced plant. One of the stumbling blocks of working with large systems is the computational burden imposed by such models. Of paramount importance to the engineer is the ability to change a parameter of the design and have the control software perform the relevant calculations in a reasonable amount of time. In order to reach this goal, one must harness the power of today's supercomputer (which will be tomorrow's personal computer) by designing new parallel algorithms. The Appendix presents a method of solving large-order Riccati equations on a shared-memory parallel vector computer such as the Cray Y-MP.

In this article we focus on controlling the solar dynamic power module of the proposed Space Station. This assembly consists of an array of mirrors that together form a concave sunlight collector/concentrator (see Fig. 1). The concentrated sunlight from the mirror assembly is converted into electricity by power electronics located in the receiver assembly. This collection method for sunlight is similar in design to that of a home satellite dish. Instead of collecting television or radio signals, the power module collects solar energy. A second key difference is that the power module must track the sun while the space station is orbiting the earth whereas a satellite dish needs only focus once on a stationary (i.e., geosynchronous) object. To maximize efficiency, both the mirror and receiver assemblies must be controlled. As one might imagine, the overall system is quite complex and consists of photovoltaic array assemblies, alpha gimbals, beta gimbals, truss and interface structures, radiators, fine pointing and tracking (FPT) actuators, concentrators, receivers, and other power conversion sub-modules. We will refer to the entire assembly as an SD module.

Two possible control options for implementing the pointing and tracking of the power modules have been analyzed. In the first option, control of an SD module is performed by two pairs of gimbals: alpha and fine pointing (FP) pitch for the elevation axis, together with beta and fine pointing yaw for the azimuth axis. A coarse pointing of an SD module is provided by the alpha and beta gimbals, while fine pointing of the concentrator assemblies is performed by the FP actuators. A sun sensor for both pitch and yaw axes is located on the concentrator to measure the angular error between the sun ray and the concentrator on the receiver with respect to the inertial axis. The pitch and yaw gyros are located on the inner and outer gimbal rings, respectively. The measurements from the sensors are then fed back to the FP controllers. In the second option, the FPT actuator for the azimuth axis is eliminated. The pointing function is achieved directly by the beta gimbal. To rotate the SD receiver and concentrator in the FP elevation axis, a linear actuator is placed between the bottom surface of the receiver and the beta gimbal assembly. FPT functions are thus provided by three gimbals: alpha and FP elevation for pitch control, and beta gimbals for the yaw control.

Preliminary specification for this system is based on the frequency separation principle to maintain system stability. The system is required to fine point the reflective surface of the solar dynamic concentrator and track the sun to a three sigma accuracy of ±0.3° per axis for the sun portion of each orbit. The preliminary control system requirements include a 0.04 Hz bandwidth for the alpha and beta axes and 0.2 Hz for the FP axis.

The focus of this article is on the control analysis results (both conventional and optimal) for the second option above.

Control Design Approach

Three different approaches can be pursued to design control algorithms for large space structures:
Direct design of a high-order optimal control algorithm based on the full-order plant model, followed by controller model reduction.

Direct design by trial-and-error of a low-order controller for the full-order plant model.

Full-order plant model reduction followed by the design of a low-order (optimal or conventional) control algorithm.

The third approach is presented in this paper, with application to the fine pointing and tracking system of the Space Station solar dynamics. Although a set of control system performances satisfying the requirements can be derived using conventional PID control laws, it is very important to gain an appreciation regarding how much improvement in the control system performance one can obtain by designing, at least "on paper," alternate optimal control algorithms.

Models for large space structures involve several hundreds, or perhaps even thousands of state variables. There are significant computational issues associated with the solution of optimal control problems for such systems. Therefore, very few results have been reported in the past concerning the practical application of optimal control algorithms for such very high-order plant models. For this reason, in most cases, the design of control algorithms for large space structures is being done by ad hoc procedures using reduced-order model representations. However, there is no guarantee for such designs that the same type of controller will perform as required or even maintain stability when the full-order, instead of the reduced-order, plant model is incorporated in the simulation. In such a situation, a series of ad hoc iterations could be pursued, without any guarantee for final success. The application of optimal control theory by considering the full-order plant model is very attractive because it can provide a stable control algorithm from the very beginning. For a completely controllable and observable plant, one can solve the associated Riccati equation and thus obtain an idealized baseline for the control solution. This approach became feasible due to recent advances using parallel processing for solving high-dimensional algebraic Riccati equations. We will present an application of this method for the optimum control design of the space station SDFPT system. Additional aspects such as order reduction and subsequent implementation of the optimal Riccati-based high-order controller will be examined in a future paper.

PID Design and Results

Effector Models: A sixth-order electromechanical actuator (EMA) model was used to represent the FP linear actuator dynamics, including the flexibility of the push rod. By selecting appropriate values for the actuator design parameters, such as the servo position gain, servo rate gain, current integrator gain, position feedback gain, overall gear ratio, and the push-rod spring constant, the EMA was designed to meet a 0.5 Hz control bandwidth specification. The sun sensor and the displacement gyro sensor models were designed to include frequencies high enough to encompass the rigid and flexible mode dynamics. These sensors were represented by a first-order model with time constant of 20 ms.

There are three independent controllers in the system, one for each gimbal. However, the structural mode shapes are interrelated. Therefore, if any disturbance is applied to one location, it will perturb and interact with the rest of the structure and, in turn, the entire control system. A proportional-integral controller with rate feedback was designed for each gimbal. For the alpha axis, the actuator, displacement sensor, and rate sensor are collocated. For the beta gimbal and the FP elevation linear actuator, the rate feedback sensors are collocated, while the position feedbacks are provided by the sun sensor located at the concentrator.

![Fig. 2. Residual error versus model reduction.](image1)

![Fig. 3. Alpha and beta gimbal angular displacements and FP actuator linear displacement.](image2)

![Fig. 4. Commanded and actual FP angular position (tracking).](image3)
The controllers were designed using the one-loop-at-a-time approach, with only the rigid body modes incorporated. The tuning process for the set of controller gains includes the error-shaping and pole-zero placement techniques, as well as the utilization of root locus plots and Bode plots to display system damping and bandwidth. The controllers were tuned to achieve 0.04 Hz rigid body control frequencies and 0.7 damping ratios for the alpha and beta gimbals. The FP elevation loop was initially designed for a bandwidth of 0.5 Hz frequency and 0.7 damping ratio.

The reduced-order models were utilized to define system stability and control performance. Eigenvalue analyses and frequency domain analyses were performed to characterize system stability. Time domain analyses were conducted to predict system transient response and control accuracy. Different control bandwidths for the beta and FP elevation axes were also investigated in order to meet system requirements. Using PID control laws with no bending filters, the system showed instability with a rigid body control bandwidth of 0.5 Hz for the FP elevation axis. Stability analyses indicated that there is a significant interaction between the alpha gimbal and FP elevation actuator. However, by detuning the FP elevation axis control bandwidth from 0.5 Hz to 0.24 Hz, the system was stabilized.

Step input command and tracing error simulations were run to predict system response time and control accuracy (Fig. 3). The beta and FP elevation axes respond to the commanded angular position in 12 s and 2 s, respectively. The FPT system is required to track the sun while orbiting the earth at a rate of 4º/min in the elevation axis. The 0.03º angular tracking error is still within the control accuracy requirement, although the FP elevation axis has been redesigned with a lower 0.24 Hz control frequency instead of 0.5 Hz (Fig. 4).

The structural dynamics mathematical models were based on the space station NASTRAN model outputs [2]. There are 14 rigid body modes and 272 elastic modes ranging from 0.07 Hz to 4.9 Hz in the full plant finite element model. Practical implementation considerations impose restrictions on the order or complexity of the control system models. To facilitate the analysis and design process, it is necessary to reduce the number of modes from the full-order models. Different reduction methods from MATLAB and COSTIN were used to generate the reduced-order models:

- The MATLAB robust control software provides a model state ranking based on the observability criterion of the system modes.
- The COSTIN program computes the approximate Hankel singular values of the full model for the FP elevation axis. Stability analyses indicated that there is a significant interaction between the alpha gimbal and FP elevation actuator. However, by detuning the FP elevation axis control bandwidth from 0.5 Hz to 0.24 Hz, the system was stabilized.

Step input command and tracing error simulations were run to predict system response time and control accuracy (Fig. 3). The beta and FP elevation axes respond to the commanded angular position in 12 s and 2 s, respectively. The FPT system is required to track the sun while orbiting the earth at a rate of 4º/min in the elevation axis. The 0.03º angular tracking error is still within the control accuracy requirement, although the FP elevation axis has been redesigned with a lower 0.24 Hz control frequency instead of 0.5 Hz (Fig. 4).

The forcing function data developed in [18] was used to study the disturbance rejection capability of the control system. Two worst-case docking simulations, representing maximum axial loads and maximum shear forces (Fig. 5) were studied. A series of load combination events as recommended in [18] was also investigated. They are docking combined with intravehicular activity (IVA) and treadmill, berthing combined with IVA and treadmill, and extravehicular activity combined with IVA and treadmill. Overall, adequate system performance in terms of stability and control accuracy was found in all given cases. The plots of disturbance response (Fig. 6) show that all angular positions measured at the sensors for the alpha, beta, and FP elevation axes are within the preliminary control accuracy requirements.

**Model Reduction**

The forcing function data developed in [18] was used to study the disturbance rejection capability of the control system. Two worst-case docking simulations, representing maximum axial loads and maximum shear forces (Fig. 5) were studied. A series of load combination events as recommended in [18] was also investigated. They are docking combined with intravehicular activity (IVA) and treadmill, berthing combined with IVA and treadmill, and extravehicular activity combined with IVA and treadmill. Overall, adequate system performance in terms of stability and control accuracy was found in all given cases. The plots of disturbance response (Fig. 6) show that all angular positions measured at the sensors for the alpha, beta, and FP elevation axes are within the preliminary control accuracy requirements.

**Model Reduction**

The structural dynamics mathematical models were based on the space station NASTRAN model outputs [2]. There are 14 rigid body modes and 272 elastic modes ranging from 0.07 Hz to 4.9 Hz in the full plant finite element model. Practical implementation considerations impose restrictions on the order or complexity of the control system models. To facilitate the analysis and design process, it is necessary to reduce the number of modes from the full-order models. Different reduction methods from MATLAB and COSTIN were used to generate the reduced-order models:

- The MATLAB robust control software provides a model state ranking based on the observability criterion of the system modes.
- The COSTIN program computes the approximate Hankel singular values of the full model for the FP elevation axis. Stability analyses indicated that there is a significant interaction between the alpha gimbal and FP elevation actuator. However, by detuning the FP elevation axis control bandwidth from 0.5 Hz to 0.24 Hz, the system was stabilized.

Step input command and tracing error simulations were run to predict system response time and control accuracy (Fig. 3). The beta and FP elevation axes respond to the commanded angular position in 12 s and 2 s, respectively. The FPT system is required to track the sun while orbiting the earth at a rate of 4º/min in the elevation axis. The 0.03º angular tracking error is still within the control accuracy requirement, although the FP elevation axis has been redesigned with a lower 0.24 Hz control frequency instead of 0.5 Hz (Fig. 4).

The structural dynamics mathematical models were based on the space station NASTRAN model outputs [2]. There are 14 rigid body modes and 272 elastic modes ranging from 0.07 Hz to 4.9 Hz in the full plant finite element model. Practical implementation considerations impose restrictions on the order or complexity of the control system models. To facilitate the analysis and design process, it is necessary to reduce the number of modes from the full-order models. Different reduction methods from MATLAB and COSTIN were used to generate the reduced-order models:

- The MATLAB robust control software provides a model state ranking based on the observability criterion of the system modes.
- The COSTIN program computes the approximate Hankel singular values of the full model for the FP elevation axis. Stability analyses indicated that there is a significant interaction between the alpha gimbal and FP elevation actuator. However, by detuning the FP elevation axis control bandwidth from 0.5 Hz to 0.24 Hz, the system was stabilized.

Step input command and tracing error simulations were run to predict system response time and control accuracy (Fig. 3). The beta and FP elevation axes respond to the commanded angular position in 12 s and 2 s, respectively. The FPT system is required to track the sun while orbiting the earth at a rate of 4º/min in the elevation axis. The 0.03º angular tracking error is still within the control accuracy requirement, although the FP elevation axis has been redesigned with a lower 0.24 Hz control frequency instead of 0.5 Hz (Fig. 4).

The structural dynamics mathematical models were based on the space station NASTRAN model outputs [2]. There are 14 rigid body modes and 272 elastic modes ranging from 0.07 Hz to 4.9 Hz in the full plant finite element model. Practical implementation considerations impose restrictions on the order or complexity of the control system models. To facilitate the analysis and design process, it is necessary to reduce the number of modes from the full-order models. Different reduction methods from MATLAB and COSTIN were used to generate the reduced-order models:

- The MATLAB robust control software provides a model state ranking based on the observability criterion of the system modes.
- The COSTIN program computes the approximate Hankel singular values of the full model for the FP elevation axis. Stability analyses indicated that there is a significant interaction between the alpha gimbal and FP elevation actuator. However, by detuning the FP elevation axis control bandwidth from 0.5 Hz to 0.24 Hz, the system was stabilized.

Step input command and tracing error simulations were run to predict system response time and control accuracy (Fig. 3). The beta and FP elevation axes respond to the commanded angular position in 12 s and 2 s, respectively. The FPT system is required to track the sun while orbiting the earth at a rate of 4º/min in the elevation axis. The 0.03º angular tracking error is still within the control accuracy requirement, although the FP elevation axis has been redesigned with a lower 0.24 Hz control frequency instead of 0.5 Hz (Fig. 4).

The structural dynamics mathematical models were based on the space station NASTRAN model outputs [2]. There are 14 rigid body modes and 272 elastic modes ranging from 0.07 Hz to 4.9 Hz in the full plant finite element model. Practical implementation considerations impose restrictions on the order or complexity of the control system models. To facilitate the analysis and design process, it is necessary to reduce the number of modes from the full-order models. Different reduction methods from MATLAB and COSTIN were used to generate the reduced-order models:

- The MATLAB robust control software provides a model state ranking based on the observability criterion of the system modes.
- The COSTIN program computes the approximate Hankel singular values of the full model for the FP elevation axis. Stability analyses indicated that there is a significant interaction between the alpha gimbal and FP elevation actuator. However, by detuning the FP elevation axis control bandwidth from 0.5 Hz to 0.24 Hz, the system was stabilized.

Step input command and tracing error simulations were run to predict system response time and control accuracy (Fig. 3). The beta and FP elevation axes respond to the commanded angular position in 12 s and 2 s, respectively. The FPT system is required to track the sun while orbiting the earth at a rate of 4º/min in the elevation axis. The 0.03º angular tracking error is still within the control accuracy requirement, although the FP elevation axis has been redesigned with a lower 0.24 Hz control frequency instead of 0.5 Hz (Fig. 4).

The forcing function data developed in [18] was used to study the disturbance rejection capability of the control system. Two worst-case docking simulations, representing maximum axial loads and maximum shear forces (Fig. 5) were studied. A series of load combination events as recommended in [18] was also investigated. They are docking combined with intravehicular activity (IVA) and treadmill, berthing combined with IVA and treadmill, and extravehicular activity combined with IVA and treadmill. Overall, adequate system performance in terms of stability and control accuracy was found in all given cases. The plots of disturbance response (Fig. 6) show that all angular positions measured at the sensors for the alpha, beta, and FP elevation axes are within the preliminary control accuracy requirements.
method to the SDFPT, the system model was modified. All the rigid body modes were removed to ensure open-loop stability. The number of sensed outputs for the square system was reduced, and a small epsilon $D$ matrix at high frequencies was also added to ensure invertibility.

Preliminary results obtained using MATLAB showed that only 50 flexible modes can be neglected (Fig. 2) out of the original 272 elastic modes in order to limit the relative error at 100%.

Summary Remarks on Modeling: Conventional control design techniques have been used to investigate feasibility of two alternative design options for the solar dynamic power modules. Using the reduced-order plant models, it has been possible to show feasibility of the PID control solution without imposing the control/structure frequency separation restriction.

During the design and analysis process, different orders of model reduction (20 modes versus 40 modes) were also investigated with the same PID controller and effector parameters. When a higher-order system model obtained by adding the next 20 significant elastic modes to the reduced-order model of 20 modes, system instability was observed, which suggested the need for further controller and/or actuator parameters retuning. According to the results from the BST method, one has to keep at least the first 272 significant modes of the plant to design a stable controller for the full-order plant model. There is no a priori guarantee that a PID controller satisfying the control requirements could be found for this more complex higher-order plant model. This suggests that for large space structures alternative design methods besides the conventional ones might be required.

Timing Results

In this section we give some data on the amount of time needed to compute the sign of the Hamiltonian matrix associated with the algebraic Riccati equation (ARE) (6). For the SDFPT module, a total of 278 modes yields a state matrix of size 556, which in turn yields a Hamiltonian matrix of size 1112. To illustrate the resources needed to calculate the sign of a matrix, we give some timing results for the symmetric algorithm derived here and the nonsymmetric algorithm presented in [20]. Table II shows the time needed to compute the sign of the Hamiltonian arising from the position and velocity control of a string of high-speed vehicles (see Example 4 in [17]. Table III shows results for the SDFPT control problem when differing numbers of elastic modes are truncated from the model. Both show that the symmetric algorithm is nearly twice as fast for large-order problems.

Interpreting timing results for parallel algorithms can sometimes be difficult. For example, parallel codes can only reduce the computation time and not the overall number of arithmetic operations. We have included two timings for each case: $t_{cpu}$ and $t_{wall}$. The first timing, $t_{cpu}$, indicates the total amount of CPU resource that is needed by the program. This figure is the sum of CPU cycles used by all processors. The second timing, $t_{wall}$, indicates the real time used (as if measured by a stopwatch or wall-clock). Since the Cray is a time-sharing operating system and the loading can vary greatly, wall-clock time can change dramatically from one run to the next. For example, the wall-clock timings for the size 1112 Hamiltonian can take as much as 3 times longer when run during peak operating hours or as little as half the time when run at 2:00 a.m., the time of lightest load. The timings reported here are realistic in the sense they are actual timings from running the jobs on a moderately-loaded system. It is interesting to note that for small problems the nonsymmetric algorithm is more efficient than is the symmetric one. This is attributed to the data motion when multiplying by $J$ which raises the factor of the $n^2$ term in this overall $O(n^3)$ algorithm.

### Table I

<table>
<thead>
<tr>
<th>Frequency (Hz)</th>
<th>COSTIN Ranking</th>
<th>MATLAB Ranking</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.6667</td>
<td>0.106112</td>
<td>1</td>
</tr>
<tr>
<td>0.6623</td>
<td>0.105411</td>
<td>2</td>
</tr>
<tr>
<td>2.9993</td>
<td>0.477367</td>
<td>3</td>
</tr>
<tr>
<td>0.5583</td>
<td>0.088859</td>
<td>4</td>
</tr>
<tr>
<td>2.6070</td>
<td>0.414929</td>
<td>5</td>
</tr>
<tr>
<td>3.3612</td>
<td>0.534967</td>
<td>6</td>
</tr>
<tr>
<td>2.4283</td>
<td>0.386487</td>
<td>7</td>
</tr>
<tr>
<td>2.4631</td>
<td>0.392026</td>
<td>8</td>
</tr>
<tr>
<td>3.9889</td>
<td>0.634872</td>
<td>9</td>
</tr>
<tr>
<td>1.0355</td>
<td>0.164810</td>
<td>10</td>
</tr>
<tr>
<td>3.8881</td>
<td>0.618829</td>
<td>11</td>
</tr>
<tr>
<td>1.0484</td>
<td>0.166863</td>
<td>12</td>
</tr>
<tr>
<td>21.162</td>
<td>3.369136</td>
<td>13</td>
</tr>
<tr>
<td>3.8296</td>
<td>0.509550</td>
<td>14</td>
</tr>
<tr>
<td>1.6187</td>
<td>0.257632</td>
<td>15</td>
</tr>
<tr>
<td>3.8525</td>
<td>0.613163</td>
<td>16</td>
</tr>
<tr>
<td>0.7304</td>
<td>0.116250</td>
<td>17</td>
</tr>
<tr>
<td>23.844</td>
<td>3.794939</td>
<td>18</td>
</tr>
<tr>
<td>24.722</td>
<td>3.934745</td>
<td>19</td>
</tr>
<tr>
<td>22.070</td>
<td>3.512653</td>
<td>20</td>
</tr>
<tr>
<td>24.347</td>
<td>3.875060</td>
<td>21</td>
</tr>
<tr>
<td>4.1983</td>
<td>0.688200</td>
<td>22</td>
</tr>
<tr>
<td>2.0167</td>
<td>0.320977</td>
<td>23</td>
</tr>
<tr>
<td>22.952</td>
<td>3.653032</td>
<td>24</td>
</tr>
<tr>
<td>24.753</td>
<td>3.935678</td>
<td>25</td>
</tr>
<tr>
<td>4.1373</td>
<td>0.650441</td>
<td>26</td>
</tr>
<tr>
<td>1.3513</td>
<td>0.215072</td>
<td>27</td>
</tr>
<tr>
<td>1.3507</td>
<td>0.214977</td>
<td>28</td>
</tr>
<tr>
<td>1.1699</td>
<td>0.186201</td>
<td>29</td>
</tr>
<tr>
<td>22.460</td>
<td>3.574725</td>
<td>30</td>
</tr>
<tr>
<td>24.110</td>
<td>3.837339</td>
<td>31</td>
</tr>
<tr>
<td>22.867</td>
<td>3.639503</td>
<td>32</td>
</tr>
<tr>
<td>4.9282</td>
<td>0.784371</td>
<td>33</td>
</tr>
<tr>
<td>4.6419</td>
<td>0.738830</td>
<td>34</td>
</tr>
<tr>
<td>27.828</td>
<td>4.429094</td>
<td>35</td>
</tr>
<tr>
<td>3.1067</td>
<td>0.494461</td>
<td>36</td>
</tr>
<tr>
<td>23.951</td>
<td>3.812032</td>
<td>37</td>
</tr>
<tr>
<td>23.350</td>
<td>3.716378</td>
<td>38</td>
</tr>
<tr>
<td>21.871</td>
<td>3.480960</td>
<td>39</td>
</tr>
<tr>
<td>25.048</td>
<td>3.986631</td>
<td>40</td>
</tr>
</tbody>
</table>

* Slight mismatch due to difference between COSTIN and MATLAB input/output
Controller Synthesis and Analysis Results

Pure LQ theory involves the use of weighting matrices $Q$ and $R$ for the state variables and control inputs to the system, respectively. The $Q$ and $R$ for the SDFPT system were selected through an iterative approach with steady-state error and system bandwidth requirements as basic guidelines. The choice of weighting matrices was also done with the reduced-order model of the plant in mind. Initially, we chose $Q$ and $R$ such that more penalty was placed on the first 40 ranked modes. The matrix $R$ was the identity while $Q$ was also an identity matrix except for the entries corresponding to the first 40 dominant modes. These entries were selected to be twice as large. The controller synthesized using this choice of $Q$ and $R$ produced steady-state errors and control bandwidth exceeding the system specifications. In a subsequent iteration, we chose $Q$ to be 100 times the original $Q$ matrix in an attempt to meet the the control system bandwidth requirement. This bandwidth became satisfactory, but the system tracking steady-state error was still outside the specifications. Several more iterations were performed using the above-described scheme in an attempt to simultaneously satisfy the control system bandwidth and steady-state error requirements. Because of the limited success in this approach, we next tried to utilize all the sensor measurement information for the optimal controller synthesis. We set $R$ to identity and $Q$ to $CTC$. In order to satisfy the requirements, we then had to vary the forward loop gain for each control axis.

Figs. 4 and 7 compare the tracking accuracy between the LQ design and the PID design presented earlier in this paper. Note that the steady-state error is within the tolerance and the LQ tracking response is much more damped than is the PID response. Fig. 8 shows the transient response of the LQ-based design. In this simulation, beta and FP axes are commanded at 0.1° while alpha is regulated at 0°. Overall, the LQ design is much more stable than the PID design. One should also note that the PID design was based on a reduced-order model and the corresponding simulations were performed on that model and not the full model that we have incorporated. In fact, when using the PID design for the full model, instability was observed.

We have presented above a very efficient numerical algorithm for the solution of high-order matrix Riccati equations. Thus, the application of optimal control for the design of large space structures becomes more feasible. It was shown on this basis that we could obtain a stable controller which met the specifications. Unlike this LQ-based design, a conventional PID controller designed using a reduced-order plant model became unstable when the full-scale plant model was simulated. For practical implementation, the full-order optimal controller should be further simplified by using, for example, an appropriate model reduction technique. This next step will be presented in a forthcoming paper.

Appendix

Numerical Solutions of Large-Scale Riccati Equations

A major difficulty with designing linear-quadratic-Gaussian (LQG) controllers for large models is solving the matrix Riccati equation that
arises from the optimal formulation. Here we have incorporated a previously-reported matrix Riccati solver that is based on a Padé approximation to the matrix sign function. We also derive a symmetric version of this algorithm for the special class of Hamiltonian matrices thereby yielding, for large problems, a near-two-fold speed increase over a previous algorithm. A typical computer run for this problem takes about six minutes, thus making large-order models a possibility for initial “paper design” studies. In this way, applicability of optimal control design for large space structures becomes practical.

This appendix first presents the optimal control algorithm, and then describes details of the numerical solution for the large-scale Riccati equation.

**Optimal Control**

As explained in the body of this article we take the approach of direct design of a high-order optimal controller based on the full-order plant model. We used the following linear plant model in the state-space formulation:

\[
\begin{align*}
\dot{x} &= Ax + Bu \\
y &= Cx
\end{align*}
\]

where \( A \in \mathbb{R}^{556 \times 556} \), \( B \in \mathbb{R}^{556 \times 6} \), and \( C \in \mathbb{R}^{12 \times 556} \). The size of the matrix \( A \) corresponds to a plant model consisting of 6 rigid body and 272 flexible modes, while the matrix \( B \) corresponds to the 6 actuators controlling the 6 rigid body modes. We note that the model is completely controllable and observable. Our controller design corresponds to the solution of the classical LQ problem, which minimizes the cost function

\[
J = \frac{1}{2} \int_0^\infty \left( x^T Q x + u^T R u \right) dt
\]

where \( Q = Q^T \geq 0 \) and \( R = R^T > 0 \) are weighting matrices of appropriate dimensions.

The optimal control which solves the above problem is well-known to be of the form

\[
u(t) = K^T x(t)
\]

where the feedback gain matrix \( K \) is

\[
K = -R^{-1} B^T X
\]

with \( X \) being the unique positive definite solution (whose existence is guaranteed by our assumptions above) of the algebraic matrix Riccati equation

\[
A^T X + XA - XBR^{-1} B^T X + Q = 0.
\]

It is also a classical result that the closed-loop matrix

\[
A + BK = A - BR^{-1} B^T X
\]

is stable.

The algorithm described here allows us to solve Riccati equations for large-scale systems in an efficient manner. Thus, the design of optimal controllers for large space systems becomes feasible.

**Matrix Sign Function**

This section describes a parallel algorithm for computing the sign function of a matrix. The sign function is useful since it can be used to compute certain invariant subspaces of a matrix. It is precisely one of these invariant subspaces which is needed to solve the Riccati equation that arises from the optimal control formulation. We have included some of the details of the parallel algorithm and discuss its implementation on a Cray Y-MP.

**Definition 1** Consider a matrix \( X \in \mathbb{C}^{n \times n} \) such that \( \sigma(X) \neq 0 \), i.e., \( X \) does not have any eigenvalues on the imaginary axis. Let

\[
S = \operatorname{sgn}(X)
\]

where \( T \) is nonsingular; \( D = \text{diag}(d_1, \ldots, d_p) \), and \( N \) is nilpotent and commutes with \( D \). Then \( S = \text{sgn}(X) \) is defined by

\[
S = T \begin{bmatrix} 
\text{sgn}(d_1) \\
\vdots \\
\text{sgn}(d_p)
\end{bmatrix} T^{-1}
\]

Further details and properties of the matrix sign function can be consulted in [20].

**Application of the Matrix Sign Function**

The matrix sign function can be used to solve the algebraic Riccati equation (6) in terms of the sign of the Hamiltonian matrix

\[
H := A - BR^{-1} B^T X
\]

associated with it. Assuming \( (A, B) \) is stabilizable and \( (C, A) \) is detectable (where \( C^T C = Q \)), then it can be shown that \( \sigma(H) = 0 \) and (6) has an admissible (symmetric, nonnegative definite, and stabilizing) solution \( X \in \mathbb{R}^{n \times n} \). The following theorem, proved in [16], shows how to compute this solution via the sign function of \( H \).

**Theorem 1** Consider the Hamiltonian matrix \( H \in \mathbb{R}^{2n \times 2n} \) defined in (10). Suppose \( \sigma(H) = 0 \) and let

\[
X = T(D + N)T^{-1}
\]
Then the admissible solution $X_0 \in \mathbb{R}^{n \times n}$ to (6) can be obtained from
\[
\begin{bmatrix} Z_1 \\ Z_2 \\ Z_1 \\ Z_2 \end{bmatrix} = \text{sgn}(H).
\]

Then the admissible solution $X_0 \in \mathbb{R}^{n \times n}$ to (6) can be obtained from
\[
\begin{bmatrix} Z_2 \\ Z_2 + I \end{bmatrix} S_x = \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix}.
\]

**Computing the Matrix Sign Function**

*Newton iteration:* Several researchers have studied and refined algorithms for computing the matrix sign function. For example, see Roberts [22], Byers [4], Bierman [3], Barraud [1], and Kenney and Laub [14]. One of the simplest methods to compute the sign of a matrix $X \in \mathbb{C}^{n \times n}$ is by the following iteration [22]:
\[
X_{n+1} = \frac{1}{2}(X_{n} + X_{n}^{-1}) X_{0} = X.
\]

Then
\[
\text{sgn}(X) = \lim_{n \to +\infty} X_{n}.
\]

This algorithm is actually Newton’s iteration to compute a square root of the identity matrix. The convergence of this algorithm is acceptable but can be improved by scaling techniques described by Kenney and Laub [15].

*Rational Iterative Methods:* In order to develop algorithms which are more amenable to parallel implementation we turn to the general rational methods of Kenney and Laub [14]. In that work a rich theory of rational recursions for the matrix sign function is presented, including Padé methods of the form
\[
X_{n+1} = X_{n} P_{k}(X_{n}^{2}) Q_{m}(X_{n}^{2}),
\]
where $P_k$ and $Q_m$ are polynomials of degree $k$ and $m$, respectively. It is also shown that the main diagonal and first subdiagonal Padé approximations are globally convergent and include inverse versions of Newton’s and Halley’s methods as special cases. Further, higher-order formulas can be shown to be equivalent to repeated Newton iterations.

Rational methods are of special interest because their partial fraction forms lead naturally to parallel algorithms. This idea has been noted elsewhere. For example, Gallopoulos and Saad [7] used this technique to develop a parallel block cyclic algorithm for solving elliptic equations. In our case, sequential matrix multiplications, which arise in evaluating the polynomials $P_k$ and $Q_m$ in (14), are avoided by using the partial fraction expansion [11],
\[
P_k(z) Q_m(z) = \sum_{i=1}^{m} w_i (z - z_i)^{-1},
\]

where $k \leq m$ and $z_1, \ldots, z_m$ are the roots of $Q_m$, which turn out to be distinct and non-repeated. In the matrix case (15) becomes
\[
P_k(X_{n}^{2}) Q_m(X_{n}^{2}) = \sum_{i=1}^{m} w_i (X_{n}^{2} - z_i)^{-1},
\]
and the individual terms in the summation can be evaluated in parallel on separate processors.

In general, the weights $w_i$ and zeros $z_i$ of a partial fraction approximation must be computed numerically. This can be a source of inaccuracy if the denominator polynomial is ill-conditioned. This problem is discussed by Gallopoulos and Saad [6] for a partial fraction approximation of the matrix exponential. Fortunately, this problem can be avoided for the sign function since the weights and zeros can be found exactly from the roots of Chebyshev polynomials. Thus, the partial fraction approach is numerically very attractive for the sign function.

The partial fraction expansion leads to algorithm-level parallelism since each fraction can be evaluated on a different processor in parallel. In order to simplify the comparison of Newton’s method with the partial fraction method, we will assume that individual processors are used to perform matrix inversions. It should be noted, however, that a multi-processor inversion routine would enhance the relative performance of Newton’s method vis-a-vis the partial fraction method.

**Implementation of a Parallel Method:** Kenney and Laub [14] showed that the iteration to compute the sign function in the scalar case can also be used in the matrix case. Hence, their results lead naturally to a parallel algorithm for computing the sign function of a matrix. For the matrix case we have the following iteration:
\[
X_{n+1} = X_0 \sum_{i=1}^{m} \frac{1}{2jm\alpha_i} \left( (X_{n} - j\alpha_i)^{-1} - (X_{n} + j\alpha_i)^{-1} \right),
\]

where $X_0 = X$, and $\alpha_i$ and $\alpha_i$ are known constants which can be precomputed as shifted zeros of Chebyshev polynomials. Then
\[
\text{sgn}(X) = \lim_{n \to +\infty} X_{n}.
\]

Note that the partial fraction expansion (17) yields complex roots and the above algorithm involves computing the inverse of $2m$ complex matrices at each step. Since the roots occur in complex conjugate pairs and because they are roots of a real polynomial, some saving can be achieved by noting that only $m$ rather than $2m$ complex matrix inversions are necessary. Another possibility is to use directly the quadratic factors that result in the complex conjugate roots. Using this idea, we can avoid complex arithmetic but at the expense of forming $X^2$. In this case, (17) becomes
\[
X_{n+1} = X_0 \sum_{i=1}^{m} \frac{1}{2jm\alpha_i} \left( (X_{n}^{2} + \alpha_i^2)^{-1} \right),
\]

where $\alpha_i \approx 0$. This becomes
\[
X_{n+1} = X_0 \sum_{i=1}^{m} \frac{1}{2jm\alpha_i} \left( (X_{n}^{2} + \alpha_i^2)^{-1} \right),
\]

The iteration (19) is terminated when the norm of the relative error is less than a user-specified tolerance $\text{tol}$, i.e.,
\[
\frac{\|X_{n+1} - X_{n}\|}{\|X_{n}\|} < \text{tol}
\]

October 1992
Symmetric Formulation for Hamiltonian Matrices: In this subsection we show how a simple rewriting of (19) leads to a symmetric version of the parallel algorithm in the case of Hamiltonian matrices.

Let $H$ denote the class of Hamiltonian matrices. Recall that $A \in \mathbb{R}^{n \times n}$ is Hamiltonian if

$$J^T A^T J = -A$$

or, equivalently, if

$$(JA)^T = JA$$

where

$$J = \begin{bmatrix} 0 & I_n \\ -I_n & 0 \end{bmatrix}.$$ 

It is also easy to show that if $A \in H$ and $A$ is nonsingular, then $A^{-1} \in H$.

This property of inverses shows that if $X_0$ is Hamiltonian and invertible, then $X_n \in H$ for all $n \geq 1$.

Now rewrite the inverse term in (19) as:

$$X_n = (X_n + \alpha I_n)^{-1} = (X_n + \alpha X_n^2)^{-1}.$$

Thus (19) becomes

$$X_{n+1} = \sum_{i=1}^{m} \frac{1}{m} (X_n + \alpha X_n^{2i})^{-1}.$$  

(21)

As this stands, we get only some slight computational advantage in that we trade two matrix multiplications for a single matrix inverse. However, the big advantage is that since $X_n$ is Hamiltonian (i.e., $J X_n$ is symmetric), one can use symmetric versions of inversion routines. This algorithm can be easily implemented on a multiprocessor machine such as a hypercube, Alliant, or Cray. Since we are using an $[m - 1, m]$ Padé approximation, we will assume that $m$ processors are available. Clearly, the bulk of the computation, which is forming the $m$ matrix inversions, can be done in parallel on $m$ processors.

For each $n$, define the symmetric matrix $Y_n = JX_n$. The following algorithm computes the sign of $X_0$:

1) For $n \geq 0$:

2) Form $Y_n^{-1}$ using symmetric inversion. Then form $\bar{Y}_n^{-1} = J(Y_n)^{-1} J$.

3) For each processor $i$, calculate $Y_i = Y_n + \alpha \bar{Y}_n^{-1}$.

4) Using symmetric inversion, form $Y_{n+1}$. Then form $\bar{Y}_{n+1} = J(Y_i)^{-1} J$.

5) Compute $Y_{n+1} = \sum_{i=1}^{m} \frac{1}{m} Y_i$.

6) If $\| Y_{n+1} - Y_n \| / \| Y_n \| < \text{tol}$, proceed to step 7; else return to step 2.

7) $\text{sgn}(X_0) = J \bar{Y}_n$.

Because of the special structure of $J$ one should note that multiplication by $J$ is just a rearrangement of the multiplied matrix. Special routines were written to accomplish these multiplications. The above algorithm was implemented on a Cray Y-MP where parallel processing is achieved by Microtasking and Macrotasking [19]. Synchronization among processors was achieved using barriers [19].

Acknowledgment

The authors wish to thank their colleagues at the NASA Lewis Research Center, in particular Ted Porada, Clint Essworth, and Jennifer Rhatigan.

References


Joseph K. Cheng has 13 years of experience in the field of mathematical modeling and computer simulation of rocket engine propulsion systems, as well as control of large space flexible structures. Currently, he is a Control Analyst at Rocketdyne, responsible for the design and analysis of Control Algorithms for the Space Station Freedom electrical power system. He is also responsible for the development of computer models, control analyses, and trade studies for various Kinetic Energy Kill Vehicle propulsion systems and components. His education includes a B.S. in mechanical engineering from the University of California at Los Angeles, a M.S. in control engineering from California State University, Northridge.

George D. Ianculescu received graduate degrees in control systems and applied mathematics from Carnegie-Mellon University in 1979 and from the University of Michigan, Ann Arbor, in 1974 and 1976, respectively. Presently he is a Principal Specialist at Rockwell International, Rockford Division. Before that he was with Litton Systems, Jet Propulsion Laboratory and E.I. DuPont. His applied aerospace research interests include system identification, modern and post-modern control algorithms, as well as computational methods for control implementation. He is a member of the Board of Governors of the IEEE Control Systems Society and also Chair of the Technical Committee on Aerospace.

Charles S. Kenney received the B.S., M.A., and Ph.D. degrees in mathematics from the University of Maryland, College Park, in 1973, 1976, and 1979, respectively. Since 1987 he has worked half time as a Numerical Analyst at the Naval Weapons Center in China Lake, CA, and half-time as a Research Engineer in the Electrical and Computer Engineering Department at the University of California, Santa Barbara. His primary research is in the field of numerical linear algebra related to control theory, especially the solution of large Riccati problems via Pade approximation theory and the matrix sign function. His work in this area has led to efficient parallel algorithms with accurate condition estimation procedures. He is a member of SIAM and IEEE.

Alan J. Laub is Chair of the Department of Electrical and Computer Engineering at UCSB and also Co-Director of the Center for Control Engineering and Computation. His research interests are in scientific computation, numerical analysis, parallel algorithms, mathematical software, computer-aided control system design, and linear and large-scale control and filtering theory. He has published numerous technical papers in these areas and his algorithms and software enjoy widespread commercial use. Dr. Laub serves or has served on the editorial boards of five major journals. He is a member of SIAM and ACM and a Fellow of the IEEE. He is a Distinguished Member of the IEEE Control Systems Society and has served the Society in numerous elected and executive capacities, including as President in 1991.

Jason H. Q. Ly received the B.S.E.E. degree with university honors from Carnegie-Mellon University in 1987, and the M.S.E.E. degree from the University of Southern California in 1989. He is currently working toward the Ph.D. degree in the USC Electrical Engineering Systems Department. Since 1987, he has been with the Control and Monitoring System Department of Rocketdyne, Rockwell International, where he has been engaged in company R&D projects, analysis, modeling and simulation of Kinetic Energy Weapons, and robust stability analysis and model reduction of the Space Station Freedom power distribution systems. He is a member of Tau Beta Pi andEta Kappa Nu.

Philip M. Papadopoulos was born in 1963 in Walnut Creek, California. He received the B.A. degree in applied mathematics from the University of California, San Diego in 1985, and the M.S. degree in mechanical engineering in 1987 from the University of California, Berkeley. He is currently a Ph.D. candidate in the Electrical and Computer Engineering Department at the University of California, Santa Barbara, and a Research Assistant in the Scientific Computation Laboratory which is part of the Center for Control Engineering and Computation (CCEC) at UCSB. His research interests are in numerical linear algebra and analysis, algorithms for ill-conditioned matrix-valued equations, methods for solving large-scale equations on shared memory and massively parallel supercomputers, and the design of general purpose scientific software.