sss & sssMOR: Analysis & Reduction of Large-Scale Dynamic Systems with MATLAB

Alessandro Castagnotto, Maria Cruz Varona, Boris Lohmann

The accurate modeling of dynamic systems often results in a large number (>10^6) of differential equations describing the evolution of the system in time. The system matrices can easily become too large for computations or even storage of state-space (ss) objects in MATLAB. In this contribution, we present two toolboxes that exploit the sparsity of large-scale systems by defining sparse state-space (sss) objects and implement both classic and state-of-the-art model reduction algorithms.

Exploiting sparsity of the system matrices

Linear time-invariant dynamical systems are often given as state-space representations. In a large-scale setting, i.e., when the order $N$ is high ($N \gg 10^6$), the matrices are generally sparse, i.e., the number of nonzero entries is small compared to $N^2$.

$$
\begin{align*}
E x &= A x + B u \\
\dot{y} &= C x + D u
\end{align*}
$$

Unfortunately, MATLAB's Control System Toolbox converts the matrices to “full”. For this reason, the definition of state-space systems by calling

$$
sys = ss(A,B,C,D)
$$

is only feasible up until an order of magnitude $10^6$. In fact, note that storing an identity matrix of size $10^6$ as “full” requires about 80GB, in the sparse case only 2.4MB!

Functionality

With sss, you can exploit sparsity when defining and manipulating dynamical systems. All you need to do is define the system as

$$
sys = sss(A,B,C,D)
$$

to start using most of the tools you are used to, like ...

Frequency domain analysis:
>> bode(sys), sigma(sys), ...

Time domain analysis:
>> impulse(sys), step(sys), ...

Further properties:
>> norm(sys), isstable(sys), ...

... as well as new functions such as
>> eigs(sys), spy(sys), diag(sys), ...

Whenever possible, these functions are adapted to exploit sparsity of sss objects.

Performance

The following table summarizes a comparison between ss/dss and sss computations:

<table>
<thead>
<tr>
<th></th>
<th>MATLAB built-in</th>
<th>sss toolbox</th>
<th>improvement factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>storage of sys</td>
<td>597.05 KB</td>
<td>24.99 KB</td>
<td>#25</td>
</tr>
<tr>
<td>bode(sys)</td>
<td>1.83 s</td>
<td>0.64 s</td>
<td>#3</td>
</tr>
<tr>
<td>sigma(sys)</td>
<td>1.67 s</td>
<td>0.97 s</td>
<td>#2</td>
</tr>
<tr>
<td>c2d(sys)</td>
<td>0.02 s</td>
<td>&lt;0.001 s</td>
<td>#20</td>
</tr>
<tr>
<td>residue(sys)</td>
<td>not feasible</td>
<td>0.12 s</td>
<td>∞</td>
</tr>
</tbody>
</table>

Notes

sss and sssMOR are open-source toolboxes distributed under GPLv2 to foster the academic exchange on software for large-scale applications and model reduction. For more info, visit www.rt.mw.tum.de?sssMOR or mail us at ssMOR@rt.mw.tum.de.

*Part of this work is supported by the German Research Foundation (DFG), Grant LO408/19-1.

Capturing the relevant dynamics with reduced order models

Even when using the sss toolbox, computations with large-scale models will be expensive. For this reason, we often seek reduced order models as good approximations of much smaller order $n < N$. For linear systems, the standard reduction framework is given by Petrov-Galerkin projections of the form

$$
E x = A x + B u \\
\dot{y} = C x + D u
$$

where the projection matrices $V_i$, $W_i$ can be computed with different methods depending on what properties of the original model should be preserved. Classical methods include modal reduction, truncated balanced realizations and rational Krylov methods, while state-of-the-art algorithms include, for instance, IRKA and CURIES SPARK.

Functionality

Model reduction in sssMOR is performed by passing an sss object of the original model to the appropriate function, together with some additional parameters.

- modalMor(sys,n)
  - Modal reduction preserving predominant eigenvalues
- tbr(sys,s0)
  - Truncated balanced realization, retaining dominant Hankel singular values
- rk(sys,sIn,sOut)
  - Rational Krylov subspace methods, matching some Taylor series coefficients of the transfer function
- irks(sys,s0)
  - Iterative rational Krylov algorithm for $\mathcal{H}_2$-optimal reduction
- cure(sys)
  - Cumulative Reduction framework with $\mathcal{H}_2$-pseudo-optimal reduction and adaptive choice of reduced order

Results

For illustration purposes, the reduction is performed on the first element of the transfer matrix only (SISO). This system can be extracted from the sss object by calling

$$
sys = sysSSMOR(1,1)
$$

All reduced models shown in the plot below are of order $n = 12$.

---

MATLAB and Control System Toolbox (Release 2015b) are registered trademarks of The MathWorks Inc., Natick, Massachusetts, United States.

SLICOT benchmark examples: http://slgic.org/20-site/126-benchmark-examples-for-model-reduction

All computations were conducted on an Intel Core i7-2640M CPU @ 2.80 GHz with 8.00 GB RAM.