



Visit the mor4ansys web page

Author: Evgenii Rudnyi, <http://Evgenii.Rudnyi.Ru/>

## ■ ModelReduction` : Manual

The goal of the package is to provide a framework for prototyping new model reduction algorithms. I find it much more convenient first to make research at the *Mathematica* level and only then plan changes in mor4ansys.

The design of the package was to reuse functions from Post4MOR as much as possible. As such, understanding of Post4MOR is required.

An example on how functions from the package can be used to perform parametric model reduction can be found at the MOR for ANSYS site (see tutorial on parametric model reduction).

## ■ Functions for Model Reduction

<code>ArnoldiProcess [nextVec, startVec, dim]</code>	gives transposed projection matrix $V$ with <code>dim</code> columns generated by the Arnoldi process with the starting vector <code>startVec</code> and the function <code>nextVec</code> to generate the next vector as a function of a previous vector.
<code>ProjectSystem [sys, V]</code>	gives a reduced dynamic system projected on the basis $V$ . <code>ProjectSystem [psys, V]</code> gives a reduced parametric system. $V$ must be the transposed projection matrix as returned by <code>ArnoldiProcess</code> .
<code>Orthogonalize [Vini, V, tol]</code>	gives a joint orthogonalized basis of $Vini$ and $V$ . <code>tol</code> is the tolerance to deflate new vectors (by default $1. \times 10^{-12}$ ).

`ArnoldiProcess` is the main function in the package. It implements the Arnoldi process and expects that a user specifies a function to compute the next vector from the previous vector. It returns the Arnoldi vectors (the projection basis) in the transposed form. The reason for this decision is that *Mathematica* stores matrices by rows and this way allows us to choose the desired number of vectors by using `Take[V, num]`.

`ProjectSystem` takes a `DynamicSystem` from Post4MOR and projects it on the low-dimensional basis found by `ArnoldiProcess`. It can also project `ParametricSystem` described belows.

`ArnoldiProcess` and `ProjectSystem` support a complex-valued `DynamicSystem`.

`Orthogonalize` merges two bases. The first basis is supposed to be orthonormal.

<code>ModelReduction [sys, dim, ops]</code>	gives a reduced dynamic system with dimension <code>dim</code> . Options are <code>ExpansionPoint</code> and options for <code>LinearSolve</code> .
---	---

`ModelReduction` is a simple shell to `ArnoldiProcess` and `ProjectSystem` to make model reduction for `DynamicSystem`. In the case of a second-order system, it uses the default strategy of MOR for ANSYS when `MatrixE` is ignored during model reduction but then projected afterward with the basis found from `MatrixM` and `MatrixE`. See papers on the MOR for ANSYS website for more detail. The expansion point by default is zero.

## ■ ParametricSystem

This is a pretty primitive container that was made to support research on parametric model reduction. Its goal is to keep an arbitrary number of system matrices and three function to convert the matrices to that required for DynamicSystem (MatrixM, MatrixE, MatrixK).

<code>MakeParametricSystem [mats, matB, matC, outNames, funs]</code>	gives a ParametricSystem. mats is a list of matrices and funs is a list of three functions that transforms mats to matM, matE and matK of DynamicSystem. For example, matM = funs[[1]][mats, par] and so on. Use Null& if you do not need a particular matrix of DynamicSystem.
<code>Matrices [psys]</code>	gives system matrices of psys.
<code>Matrix [psys, i]</code>	gives the i-th system matrix of psys.
<code>MatrixB [psys]</code>	gives matB of psys.
<code>MatrixC [psys]</code>	gives matC of psys.
<code>OutputNames [psys]</code>	gives outNames of psys.
<code>TransformFunctions [psys]</code>	gives funs of psys.
<code>MatrixMFunction [psys]</code>	gives the tranformation matrix to obtain matM= MatrixMFunction [psys] [mats, par]
<code>MatrixEFunction [psys]</code>	gives the tranformation matrix to obtain matE= MatrixEFunction [psys] [mats, par].
<code>MatrixKFunction [psys]</code>	gives the tranformation matrix to obtain matK= MatrixKFunction [psys] [mats, par].

One can use parametric system by converting it to DynamicSystem for particular parameter values and then using StationarySolution, TransientSolution or HarmonicSolution.

<code>MakeDynamicSystem [psys, {par}]</code>	gives DynamicSystem obtained from psys at values of parameters specified by a list.
--	---

ProjectSystem describe above can be used for ParametricSystem as well. In this case, it returns projected (low-dimensional) ParametricSystem.