— M-M.E.S.S.-3.2 —
The “Matrix Equations, Sparse Solvers” toolbox for MATLAB® and GNU Octave

J. Saak§

§ Max Planck Institute for Dynamics of Complex Technical Systems, Sandtorstr. 1, 39106 Magdeburg, Tel: ++49(0)391/6110216, E-Mail: saak@mpi-magdeburg.mpg.de

M-M.E.S.S. provides low rank solvers for large scale symmetric matrix equations with sparse or sparse + low rank coefficients. The main focus is on differential and algebraic Riccati equations appearing in control and model order reduction, as well as algebraic Lyapunov equations for, e.g., balanced truncation.

The underlying dynamical system may be governed by ordinary differential equations of first or second order or structured proper differential-algebraic equations (DAEs) that allow for implicit index reduction.

The solvers philosophy is to always work on the implicitly linearized (for second order systems) and/or implicitly projected (in the DAE case) matrix equations. That means, the implicit Lyapunov or Riccati equation is always of the form known for a standard first-order ODE, that may have a non-identity but invertible E matrix,

$$\begin{aligned} E\dot{x}(t) &= Ax(t) + Bu(t), \\ y &= Cx(t) + Du(t). \end{aligned}$$

In close relation to the predecessor LyaPack, we use user-supplied functions that implement the actions of the system matrices E and A in multiplication and (shifted) solves. We provide those functions for standard state-space systems, second-order systems, structured DAEs of index 1 and 2, as well as second-order DAEs of indices 1, 2 and 3.

Recent additions have especially made the Riccati solvers more powerful. We now provide direct support for the mos general LQR Riccati equations of the form

$$0 = A^{\mathsf{T}}XE + E^{\mathsf{T}}XA + C^{\mathsf{T}}QC - \left(E^{\mathsf{T}}XB + S\right) R^{-1} \left(B^{\mathsf{T}}XE + S^{\mathsf{T}}\right),$$

with Q, R, and S the weight matrices from the quadratic cost functional. Allowing indefinite R, we can now also solve the equations required for model order reduction using

Positive real balancing

$$\begin{aligned} 0 &= APE^{\mathsf{T}} + EPA^{\mathsf{T}} + (EPC^{\mathsf{T}} - B)(D + D^{\mathsf{T}})^{-1}(EPC^{\mathsf{T}} - B)^{\mathsf{T}} \\ 0 &= A^{\mathsf{T}}QE + E^{\mathsf{T}}QA + (E^{\mathsf{T}}QB - C^{\mathsf{T}})(D + D^{\mathsf{T}})^{-1}(E^{\mathsf{T}}QB - C^{\mathsf{T}})^{\mathsf{T}} \end{aligned} \qquad \text{(PRARE)}$$

Bounded real balancing

$$\begin{aligned} 0 &= APE^{\mathsf{T}} + EPA^{\mathsf{T}} + BB^{\mathsf{T}} + (EPC^{\mathsf{T}} + BD^{\mathsf{T}})(I - DD^{\mathsf{T}})^{-1}(EPC^{\mathsf{T}} + BD^{\mathsf{T}})^{\mathsf{T}} \\ 0 &= A^{\mathsf{T}}QE + E^{\mathsf{T}}QA + C^{\mathsf{T}}C + (E^{\mathsf{T}}QB + C^{\mathsf{T}}D)(I - D^{\mathsf{T}}D)^{-1}(E^{\mathsf{T}}QB + C^{\mathsf{T}}D)^{\mathsf{T}} \end{aligned} \qquad \text{(BRARE)}$$

Linear-quadratic Gaussian balancing

$$\begin{aligned} 0 &= APE^{\mathsf{T}} + EPA^{\mathsf{T}} + BB^{\mathsf{T}} - (EPC^{\mathsf{T}} + BD^{\mathsf{T}})(I + DD^{\mathsf{T}})^{-1}(EPC^{\mathsf{T}} + BD^{\mathsf{T}})^{\mathsf{T}} \\ 0 &= A^{\mathsf{T}}QE + E^{\mathsf{T}}QA + C^{\mathsf{T}}C - (E^{\mathsf{T}}QB + C^{\mathsf{T}}D)(I + D^{\mathsf{T}}D)^{-1}(E^{\mathsf{T}}QB + C^{\mathsf{T}}D)^{\mathsf{T}} \end{aligned} \tag{LQGARE}$$

Additionally, we have refactored the code to be more extensible and readable, and our test setup to allow for more energy-efficient software development. We have deprecated the support for model order reduction, moving all development of those routines to the MORLAB package, which since version 6.0 now supports sparse systems using M-M.E.S.S. as its solver backend.