DMDSP: Software for Sparsity-Promoting Dynamic Mode Decomposition

Mihailo R. Jovanović*
Department of Electrical and Computer Engineering,
University of Minnesota, Minneapolis, MN 55455, USA

Peter J. Schmid†
Laboratoire d'Hydrodynamique (LadHyX), Ecole Polytechnique, 91128 Palaiseau cedex, France

Joseph W. Nichols‡
Department of Aerospace Engineering and Mechanics,
University of Minnesota, Minneapolis, MN 55455, USA
(Dated: January 8, 2014)

We provide a brief description of a MATLAB implementation of the Sparsity-Promoting Dynamic Mode Decomposition (DMDSP) algorithm. Additional information about the examples considered in the paper, along with MATLAB source codes and problem data, can be found at:

`http://www.ece.umn.edu/users/mihailo/software/dmdsp/`

* http://umn.edu/home/mihailo/; mihailo@umn.edu
† http://yakari.polytechnique.fr/people/peter/; peter.schmid@ladhyx.polytechnique.fr
‡ http://www.aem.umn.edu/people/faculty/bio/nichols.shtml; jwn@umn.edu

I. DESCRIPTION OF MATLAB FILES AND PROBLEM DATA

- Matlab files

  dmdsp.zip – contains all Matlab functions and problem data required to run DMDSP

- Description of m-files

dmdsp.m	–	an ADMM-based sparsity-promoting DMD algorithm;
run_dmdsp.m	–	explains how to run dmdsp;
snapshots_channel.m	–	generates matrix of snapshots for the channel flow example;
plots.m	–	plots figures shown in the paper and on the website.

A. Description of dmdsp.m

- Matlab syntax

  answer = dmdsp(P,q,s,gammaval,options);

- Description: Matlab function dmdsp.m takes the problem data $\{P, q, s, \gamma\}$ and the input options and returns the γ-parameterized family of solutions to both the sparsity-promoting DMD problem

$$\underset{\alpha}{\text{minimize}} \;\; J(\alpha) \; + \; \gamma \sum_{i\,=\,1}^{r} |\alpha_i| \tag{SP}$$

  and the constrained quadratic optimization problem

$$\begin{array}{ll} \underset{\alpha}{\text{minimize}} & J(\alpha) \\ \text{subject to} & E^T \alpha \; = \; 0. \end{array} \tag{POL}$$

- Input options allows users to specify the following parameters:
  - options.rho – augmented Lagrangian parameter ρ;
  - options.maxiter – maximum number of ADMM iterations;
  - options.eps_abs – absolute tolerance;
  - options.eps_rel – relative tolerance.
- If options argument is omitted, the default values are set to:
  - options.rho = 1;
  - options.maxiter = 10000;
  - options.eps_abs = 1.e-6;
  - options.eps_rel = 1.e-4;
- The γ-parameterized output answer is a structure that contains
  - answer.gamma – sparsity-promoting parameter γ;
  - answer.xsp – vector of amplitudes α resulting from the optimization problem (SP);
  - answer.xpol – optimal (polished) vector of amplitudes α resulting from the structured quadratic optimization problem (POL);
  - answer.Jsp – least-squares residual resulting from the optimization problem (SP);
  - answer.Jpol – optimal (polished) least-squares residual resulting from the optimization problem (POL);
  - answer.Ploss – optimal (polished) performance loss $\%\,\Pi_{\text{loss}} \; := \; 100\,\sqrt{J(\alpha_{\text{pol}})/J(0)}$;
  - answer.Nz – number of nonzero elements in the vector of amplitudes α_{pol}.

B. Description of run_dmdsp.m

- Matlab syntax

  [Fdmd,Edmd,Ydmd,xdmd,answer] = run_dmdsp;

- Matlab function run_dmdsp.m allows users to:
  - run standard DMD algorithm;
  - call dmdsp.m.

  When prompted to select the flow type, please choose:
  - 1 – for channel flow;
  - 2 – for screeching jet;
  - 3 – for cylinder bundle.

  You will also be prompted to enter a desired number of grid points for the spatially-promoting parameter γ; the default value is set to 200. These points are logarithmically spaced between minimal and maximal values that are well-suited for the three examples presented in our paper and on this website.

  run_dmdsp.m will then upload data necessary for conducting DMD and DMDSP and return:
  - Fdmd – optimal matrix on the subspace spanned by the POD modes U of Ψ_0;
  - Edmd – eigenvalues of F_{dmd};
  - Ydmd – eigenvectors of F_{dmd};
  - xdmd – optimal vector of DMD amplitudes;
  - answer – γ-parameterized structure containing the output of dmdsp.m.

C. Description of snapshots_channel.m

- Script snapshots_channel.m explains how to generate matrices of snapshots for the channel flow example. After these have been obtained, the function run_dmdsp.m allows users to:
  - run standard DMD algorithm;
  - call dmdsp.m

D. Description of plots.m

Matlab script plots.m allows users to reproduce figures shown in the paper and on

`http://www.ece.umn.edu/users/mihailo/software/dmdsp/`

E. Description of problem data

We provide users with problem data that will allow them to reproduce our results (these are stored in the respective .mat files in channel, screech, and cylinder subdirectories). The channel flow example is used to guide users through all steps required for conducting both DMD and DMDSP, including collection of the matrices of snapshots from numerically generated samples. In order to run snapshots_channel.m, several files from A Matlab Differentiation Matrix Suite are provided in subdirectory channel. For the other two examples, the matrices $U^* \Psi_1$, Σ, and V are provided (these are minimal requirements for conducting DMD and DMDSP). For the screeching jet example the matrices of snapshots Ψ_0 and Ψ_1 are simply too large; each snapshot requires 256Mb of storage in double precision format. To handle such large matrices, DMD was implemented using a MapReduce framework so that the matrices Ψ_0 and Ψ_1 could be stored and processed across several storage discs. For the cylinder bundle example the experimental snapshots were generated by researchers at Electricite de France and we have not acquired permission to make these publicly available.