The code for regression-based dimensionality reduction (RDR) may be run as-is but substantial speedups can be achieved using the mmx package the performs fast matrix operations over loops using .mex functions. Unfortunately, the setup for getting the mmx package up and running is less than straightforward. These instructions will walk you through the process. I currently only have setup instructions for mac users. I developed these instructions using Matlab R2014b and it is possible that the setup may have changed since then (I have gotten some feedback to that effect). I will do my best to keep these instructions up-to-date. I am working on setup instructions for other platforms and will post them soon.

mmx setup instructions for mac (Matlab R2014b):

Getting mmx functions to work in Matlab for mac:

If you are not yet set up for compiling .mex files with Xcode:

1) Make sure you have the latest version of Xcode. This is necessary for compiling the mex files.

a. If not, download the latest version from the app store

2) Matlab will not recognize Xcode as a compiler if the versions of either Matlab or Xcode are too recent. A patch has been made by Mathworks:

a. Download the patch (zip file at the bottom of the accepted answer) from here

b. Follow the instructions provided by the mathworks support team. This is mostly copying some commands into the terminal that they provide for you.

Get the Intel MKL Library

1) You can download it for free from here once you have registered. The registration is pretty short.

Get the mmx package

1. You will need to download the mmx package from the Mathworks file exchange from here
2. Once you have set up Xcode as your compiler (following instructions above), downloaded the mmx package, and installed the MKL library, follow the instructions in the comments of the build_mmx.m function, found in the mmx_package directory, reproduced here for convenience:

Run the following commands in Linux/Mac terminal:

```
sudo -s
cd /opt/intel/mkl/tools/builder
cat blas_example_list > blas_lapack_list
cat lapack_example_list >> blas_lapack_list
```

For Linux 64 bit:

```
make libintel64 interface=ilp64 export=blas_lapack_list
name=libsingle_mkl_ilp64 threading=sequential
```
For Linux 32 bit:
```
make libia32 interface=lp64 export=blas_lapack_list
name=libsingle_mkl_32 threading=sequential
```

For Mac:
```
make libuni interface=ilp64 export=blas_lapack_list
name=libsingle_mkl_ilp64 threading=sequential
```

A new libsingle_mkl_ilp64.so, libsingle_mkl_32.so, or libsingle_mkl_ilp64.dylib will appear. This needs to be copied to Matlab's external libraries directory.

For Mac:
```
cp libsingle_mkl_ilp64* MATLAB_ROOT/extern/lib/maci64
```

For Linux 64 bit:
```
cp libsingle_mkl_ilp64* MATLAB_ROOT/extern/lib/glnxa64
```

For Linux 32 bit:
```
cp libsingle_mkl_32* MATLAB_ROOT/extern/lib/glnx86
```

Where MATLAB_ROOT is the installation directory of your Matlab.

3. Run mmx_package.m