

A Brief Guide for Using ADMAT 2.0 Student

© 2009-2010 Cayuga Research Associates, LLC

This guide provides a brief introduction to the use of ADMAT 2.0 Student. Please consult Section 5.4 and 5.5 in our User's Guide for a detailed discussion on all functionalities and usage of ADMAT 2.0 Student (henceforth referred to as ADMAT for short).

ADMAT belongs to the “operator overloading” class of *automatic differentiation* tools and uses object-oriented programming features in MATLAB. Thus, ADMAT requires MATLAB 6.5 or above.

1. ADMAT Installation and Activation

1.1 Installation Instructions for Windows Users

ADMAT is provided in a zip file **ADMAT-2.0.zip**. Place the zip file in a proper directory and unzip it. There are two methods for setting the MATLAB search paths for ADMAT.

Method 1:

1. Click “File” in MATLAB window.
2. Choose the “Set Path” option.
3. Click the “Add with Subfolders” button.
4. Find the directory for ADMAT in the “Browse for Folders” window and click “OK”.
5. Click the “Save” button to save the paths for ADMAT and click the “Close” button.
6. Type “startup” at the MATLAB prompt or exit MATLAB and re-start MATLAB.

Method 2:

Access the ADMAT directory. Edit the **startup.m** file to add all subdirectories of ADMAT to the MATLAB search path manually. Type “startup” at MATLAB prompt to set up the paths for ADMAT.

There is a “success” message if ADMAT is properly installed.

1.2 Installation Instructions for Unix or Linux Users

Unzip **ADMAT-2.0.zip** using “unzip ADMAT-2.0.zip” at the Unix or Linux prompt; follow Method 2 described above.

1.3 ADMAT 2.0 Activation

After the proper installation of ADMAT 2.0, activation of ADMAT 2.0 may be needed in a newly-launched MATLAB session.

If the message ‘ADMAT 2.0 installed successfully’ appears in the MATLAB command window at the start of a new MATLAB session, no activation of ADMAT 2.0 is needed. ADMAT 2.0 is ready for use.

Otherwise, first make sure that ADMAT 2.0 has been properly installed. Then make the folder for

ADMAT 2.0 as the current folder of the MATLAB command window; type “startup” to activate ADMAT 2.0. The message ‘ADMAT 2.0 installed successfully’ should appear in the MATLAB command window, which indicates that ADMAT 2.0 is now activated and ready for use.

2. First Order Derivative Computation

2.1 Forward Mode AD

Using the forward mode AD provided in ADMAT, a user can easily compute the first derivatives of functions that are defined using arithmetic operations and intrinsic functions of MATLAB. Users can define their own MATLAB functions as usual. There is no restriction on the number of input arguments a user-defined function may have. If there is more than one input argument, the derivative with regard to any input argument can be computed by the forward mode AD without any change to the function definition.

In this section, we will give an example on the use of the forward mode AD for computing the Jacobian matrix of Broyden function at point $x = [1 \ 1 \ 1]$. Users can find more examples in Section 5.4 of the User’s Guide.

1. Create a **deriv** class object.

```
>> x = deriv([1,1,1], eye(3))
```

```
val =  
    1    1    1  
deriv =  
    1    0    0  
    0    1    0  
    0    0    1
```

2. Evaluate the Broyden function at point x .

```
>> y = broyden(x)
```

```
val =  
    0   -1    1  
deriv =  
   -1   -2    0  
   -1   -1   -2  
    0   -1   -1
```

3. Get the value of the Broyden function at point x .

```
>> yval = getval(y)
```

```
yval =  
    0   -1    1
```

4. Get the Jacobian matrix of the Broyden function at point x .

```
>> J = getydot(y)
```

```
J =  
-1 -2 0  
-1 -1 -2  
0 -1 -1
```

2.2 Reverse mode AD

The reverse mode AD provided in ADMAT uses a virtual tape to record all the intermediate values and operations performed in the function evaluation. Computation starts from the end of the tape; a MATLAB global variable is used to go backward through the tape. The computed derivative is finally recorded at the beginning of the tape. In this section, we will give an example of computing the first order derivative by the reverse mode AD.

The following example shows how to compute the transpose of the Jacobian matrix J^T of the Broyden function at point $x = [1\ 1\ 1]$ by using the reverse mode AD.

1. Create a **derivtape** class object.

```
>> x = derivtape([1,1,1], 1)
```

```
val =  
1 1 1  
varcount =  
1
```

2. Evaluate the Broyden function at point x .

```
>> y = broyden(x)
```

```
val =  
0  
-1  
1  
varcount =  
40
```

3. Get the value of the Broyden function at point x .

```
>> yval = getval(y)
```

```
yval =  
0  
-1  
1
```

4. Get the transpose of the Jacobian of the Broyden function at point x .

```
>> JT = parsetape(eye(3))
```

```
JT =  
-1 -1 0  
-2 -1 -1  
0 -2 -1
```

3. Troubleshooting

Below we list several potential problems that may occur in the use of ADMAT.

1. ??? Conversion to double from deriv is not possible.

This usually means a **deriv** class object is assigned to a double class variable. Check both sides of the assignment statement and make sure that they are of the same data type.

2. ??? Error using ==> XXX

Function XXX has not been defined for variables of class **deriv**. A number of MATLAB functions have not been overloaded in ADMAT yet. Please contact Cayuga Research for extending ADMAT to the MATLAB functions of your interest.

3. ??? Undefined function or variable **deriv**

ADMAT has not been installed yet. Please make sure that ADMAT is properly installed.

4. ??? Error using ==> deriv/deriv

Please restart ADMAT. There may be a license problem.

ADMAT detects a possible license error. Please restart ADMAT.

5. The ADMAT 2.0 license has expired. Please contact Cayuga Research for a license extension.

This indicates that the license for ADMAT 2.0 has already expired. Please contact Cayuga Research for license renewal.

6. Do not use Matlab command **clear all** to clear your workspace while using ADMAT. This would remove all ADMAT global variables from memory: unpredictable errors may then occur. Instead, use **clear** selectively as needed.

7. ADMAT 2.0 only performs 1-D and 2-D matrix operations. In other words, it cannot perform 3-D or higher-dimension operations.

8. The computed derivatives are incorrect. Please check the following issues.

- The command **clear all** should not be called while using ADMAT.
- The data type of the dependent variable must be consistent with that of the input independent variable in a user-defined function (See Section 3.4 in the User's Guide for details).

Finally, if there is still an error, please contact Cayuga Research for help.