UniDyn--Demo-01.nb

John A. Marohn jam99@cornell.edu Cornell University

Abstract: This demonstration notebook loads the **UniDyn** package and executes the package's unit tests.

Set the path to the package

Tell *Mathematica* the path to the directory containing the packages. For the \$NCPath variable, put the directly where the /NC folder is installed; the \$NCPath name should not end with /NC.

EDIT THE FOLLOWING PATH STRINGS:

```
In[217]:= $NCPath = "/Users/jam99/Dropbox";
$UniDynPath =
    "/Users/jam99/Dropbox/MarohnGroup__Software_Library/UniDyn/
    unidyn";
```

YOU SHOULD NOT NEED TO EDIT ANYTHING FROM HERE ONWARDS.

Load the package

Append the package path to the system path. Before trying to load the package, ask *Mathematica* to find it. This is a test that we directed *Mathematica* to the correct directory. The output of this command should be the full system path to the Uni-Dyn.m file.

```
In[219]:= $Path = AppendTo[$Path, $NCPath];
    $Path = AppendTo[$Path, $UniDynPath];
    FindFile["UniDyn`"]
    FindFile["NC`"]
Out[221]= /Users/jam99/Dropbox/MarohnGroup__Software_Library/UniDyn/unidyn/UniDyn.m
Out[222]= /Users/jam99/Dropbox/NC/init.m
```

Now that we are confident that the path is set correctly, load the package. Setting the global \$VerboseLoad variable to True will print out the help strings for key commands in the package.

```
In[223]:= $VerboseLoad = True;
    Needs["UniDyn`"]
```

Execute the units tests in batch

Included with the package are a number of files, ending in "-tests.m", that contain tests of the package's functions -- so-called unit tests. Set the working directory to the package directory and pretty-print the directory name.

```
In[225]:= SetDirectory[$UniDynPath];
     TableForm[{{$UniDynPath}}, TableHeadings → {None, {"Directory"}}]
Out[226]//TableForm=
     Directory
     /Users/jam99/Dropbox/MarohnGroup__Software_Library/UniDyn/unidyn
```

Get the names of all the unit-testing files included with the package (following my convention that the unit testing file end in "-tests.m"). Pretty-print the names of the unit-test files included with the package.

```
In[227]:= fn = FileNames["*-tests.m"];
     TableForm[{{fn}}, TableHeadings → {None, {"Test files found"}}]
Out[228]//TableForm=
     Test files found
      Comm-tests.m
     Evolve-tests.m
     Mult-tests.m
     OpCreate-tests.m
     Osc-tests.m
      Spins-tests.m
```

Finally, carry out the unit tests.

In[229]:= test\$report = TestReport /@ fn; TableForm[Table[test\$report [[k]], {k, 1, Length[test\$report]}]]

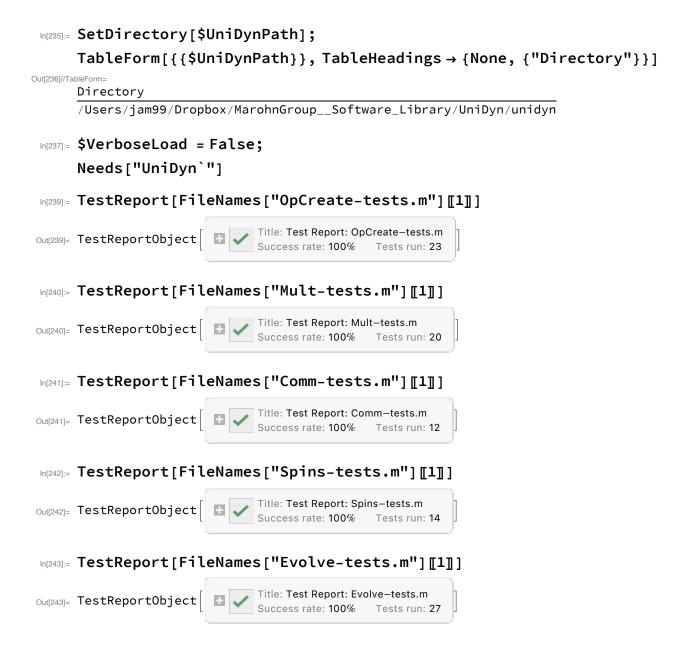


Make a report.

```
In[231]:= tests$passed$total = Plus @@ (test$report[#]["TestsSucceededCount"] & /@
          List @@ Table[k, {k, 1, Length[test$report]}]);
     tests$failed$total = Plus @@ (test$report[#]["TestsFailedCount"] & /@
          List @@ Table[k, {k, 1, Length[test$report]}]);
    Print[Style[ToString[tests$passed$total] <> " tests passed",
       FontWeight → Bold, FontSize → 18, FontColor → Blue]]
     Print[Style[ToString[tests$failed$total] <> " tests failed",
       FontWeight → Bold, FontSize → 18, FontColor → Red]]
     116 tests passed
     0 tests failed
```

Execute the units tests one-by-one

Re-execute the tests in an order determined by us. This is useful for debugging. Running the *Evolve-test.m* file takes a minute.



Congratulations

At this point you should have

- (1) loaded the NCAlgebra and UniDyn packages and
- (2) run the UniDyn units tests demonstrating that UniDyn is working as expected.