ACADO installation

Mario Zanon, Rien Quirynen and Moritz Diehl

ACADO Toolkit \(^1\) is a software environment and algorithm collection for automatic control and dynamic optimization. It provides a general framework for using a great variety of algorithms for direct optimal control, including model predictive control, state and parameter estimation and robust optimization. ACADO Toolkit is implemented as self-contained C++ code and comes along with a user-friendly MATLAB interface.

The installation procedure is OS dependent. For the installation instructions please refer to https://sourceforge.net/projects/acado/

Configuring MATLAB: Once a compiler is installed it needs to be linked to MATLAB. Open MATLAB (a recent version of MATLAB is required) and run in command window:

\[mex\ -setup;\]

MATLAB returns:

Please choose your compiler for building external interface (MEX) files:
Would you like mex to locate installed compilers [y]/n?

Type "y" and hit enter. MATLAB then shows you a list of installed compilers. Enter the number corresponding to the GCC compiler (in this case 1) and hit enter.

The options files available for mex are:

1: /software/matlab/2009b/bin/gccopts.sh :
Template Options file for building gcc MEX-files

2: /software/matlab/2009b/bin/mexopts.sh :
Template Options file for building MEX-files via the system ANSI compiler

0: Exit with no changes

Enter the number of the compiler (0–2):

Confirm the result by writing "y" and hitting enter.

\(^1\)https://sourceforge.net/projects/acado/
Building the ACADO interface: Open Matlab in the `<ACADO_ROOT>` directory. Then navigate to the MATLAB installation directory by running:

```bash
cd interfaces/matlab/;
```

You are now ready to compile the ACADO interface. This compilation will take several minutes but needs to be done only once. Run ”make” in your command window:

```bash
make clean all;
```

You will see:

```
Cleaning up all ACADO files...
Removing ACADO folders from Matlab path...
Clean completed.
Making ACADO...
```

and after a while, once the compilation is finished, you will see something like:

```
ACADO successfully compiled.
Needed to compile 249 file(s).
If you need to restart Matlab, run this make file again
to set all paths or run savepath in your console to
save the current search path for future sessions.
```

ACADO has now been compiled. As the text indicated, every time you restart MATLAB you need to run ”make” again to set all paths. When running ”make” again no new files need to be compiled and the process will only take a few seconds. However, it is easier to save your path for future Matlab sessions. Do so by running ”savepath” in your command window:

```bash
savepath;
```

Running your first example: We will now run a simple ”getting_started” example that will auto generate an ACADO integrator with sensitivities. First go to the following examples folder:

```bash
cd examples/code_generation/simulation/
```

Now type ”getting_started” in the command window to run the example:

```bash
getting_started
```

The output in the command window should consist of only the following parts:

1. ACADO translates the Matlab script into ACADO C++ syntax and compiles it:

```
Writing c++ files...
Compiling c++ files...
```

2. the resulting file is run, which will call ACADO to auto generate the requested code in the folder ”getting_started_export”: 

```bash
```
3. the auto generated code is then compiled into MEX files to be used from MATLAB:

```
COMPILATION OF MEX FILES ...
  mex COPTIMFLAGS='−DNDEBUG −O3' −output integrate.mexa64
     getting_started_export/integrate.c
     getting_started_export/integrator.c
  mex COPTIMFLAGS='−DNDEBUG −O3' −output rhs.mexa64
     getting_started_export/rhs.c
     getting_started_export/integrator.c
```

4. the rest of the output reports some accuracy and timing results after comparing a simulation with the auto generated ACADO integrator and one with the MATLAB integrator ‘ode45’