Manual for the Installation of OpenBlas and Custom Functions for Vensim

Introduction:

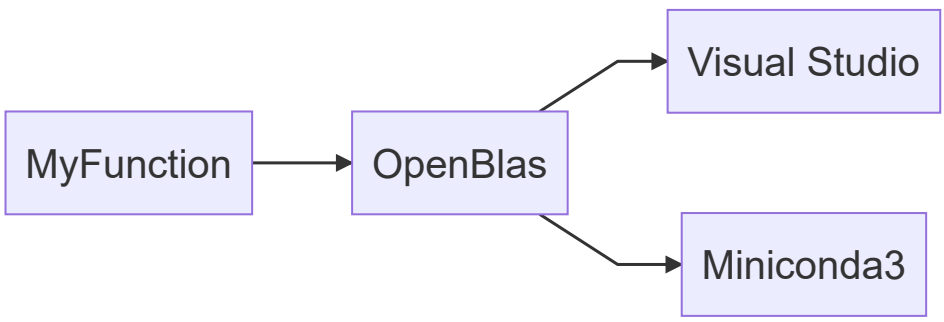
OpenBlas is a library made for optimizing the execution of mathematical operations, mainly matrix related ones. Its installation in a specific hardware configures the library so that it uses hardware specific optimizations, including parallel processing and certain CPU exclusive operations among many other minute optimizations.

This lets us execute complex mathematical operations faster than Vensim can, mainly because Vensim only uses one CPU core while OpenBlas can use more cores, as long as they are available, and use them more efficiently.

OpenBlas is used by the library that Vensim loads our custom functions from, and thus, it is necessary in order for it to function properly, among other dependencies OpenBlas requires. These dependencies are listed in this manual and they are part of the software used to install OpenBlas.

Since OpenBlas is a x64 library, this library is x64 too, so Vensim 8 or newer is required to use the custom functions.

In order to install OpenBlas, you need to install other software before, which are Miniconda 3 and any version of Microsoft Visual Studio 2015 or newer, having installed the desktop development with c++ package. This guide will cover how to install these programs too.



Miniconda 3 is a terminal that will be used to download some of the libraries OpenBlas uses and to ensure it gets installed in the correct environment.

Microsoft Visual Studio needs to be installed so that we can use some of its files for the installation. There is no need to sign in, just having it installed is enough.

The steps we will be following can be found on the official OpenBlas installation guide for windows, in the first section (`1. Native (MVSC) ABI`). However, you need to pay extra attention to certain steps in order to be able to use OpenBlas in Vensim, specifically the directories you decide to install the software in. The list of the dependencies and a command that opens a file from Visual Studio will assume that you have installed the software in the default directory.

This guide also includes the commands to install Miniconda and Visual Studio through chocolatey in case you want to do so. This is completely optional.

Miniconda 3 installation:

Miniconda can be downloaded from this link, scrolling down until you get to the section "Latest Miniconda Installer Links". The version used in the making of the library is the one for python 3.9, 64 bits. However, any version should work as long as it is 64 bits. Older version links are below the "Latest Miniconda Installer Links" section, grouped by OS.

The installation is really straightforward, follow the instructions from the installer. However, it is really important that you note the directory you install Miniconda in. You may need to run the installer as an administrator.

You can also install it through chocolatey with the command: `choco install miniconda3`

Microsoft Visual Studio installation:

Microsoft Visual Studio can be downloaded from microsoft's page. The recommended version to install is 2019, but any newer than 2015 included should work. Follow the steps of the installer. However, when prompted to select the packages to install, be sure to have "desktop development with c++" selected before continuing.

There usually are three options for downloading any version of Visual Studio: Community, Professional and Enterprise. The Community version is free and has everything we need for the installation.

In case you already had Visual Studio installed but without the package, you can just run the installer again, click modify and you will be directed to the package selection screen, where you can add new packages. Select and install the desktop development with c++ if that is the case.

You can also install the base Visual Studio with the command `choco install visualstudio2019community`, then the package with `choco install visualstudio2019-workload-nativedesktop`.

OpenBlas installation:

This next couple of steps lead to the installation itself. There will be a command that may take an hour or longer, depending on hardware. It is advised to continue installing only if you have enough time.

First get the binary packages from this link. The version used for the development of the library is 3.15, however, downloading the latest stable version available is recommended.

Downloading the latest version is as straightforward as clicking the "Download Latest Version" green button. However, to download any specific version you have to click on the folder named after the version first, then click on the "OpenBLAS x.x.xx version.zip" file to start the download.

Then unzip it and enter in the folder which was in the zip. You can know if you are in the right directory if you can find a file named “Makefile”. This directory is the one you will have to go to in the next step, when using the Miniconda command prompt.

Open the Miniconda 3 command prompt, which can be easily found by searching “Anaconda Prompt” or “miniconda3” in the windows search bar, then use ‘cd’ to navigate to the directory mentioned above, the one where OpenBlas is.

Now we need to copy and paste the following commands one by one. For an explanation on what each command does, check the OpenBlas installation guide.

```
conda update -n base conda
conda config --add channels conda-forge
conda install -y cmake flang clangdev perl libflang ninja
"c:/Program Files (x86)/Microsoft Visual
Studio/2019/Community/VC/Auxiliary/Build/vcvars64.bat"
```

To know if this command has worked properly, type the command `link` in the console. If it doesn’t return “command not found” or something similar, it has worked properly.

If this last command hasn’t worked, it may be because you don’t have Visual Studio installed.

If you don’t have Visual Studio 2019, but instead have a different year edition, change the 2019 in the command by the version of the Visual Studio you have installed.

Also, be sure to include the quotation marks when entering the command.

It may be possible to use a version prior to 2015, you can check this guide if that’s the case.

The next set of commands are the ones that need you to be in the correct directory. If you get an error related to not being in the OpenBlas directory, you need to run all commands from this point onwards again, but there is no need to run the previous ones again.

```
set "LIB=%CONDA_PREFIX%\Library\lib;%LIB%"
set "CPATH=%CONDA_PREFIX%\Library\include;%CPATH%"
mkdir build
cd build
cmake .. -G "Ninja" -DCMAKE_CXX_COMPILER=clang-cl -DCMAKE_C_COMPILER=clang-cl
-DCMAKE_Fortran_COMPILER=flang -DCMAKE_MT=mt -DBUILD_WITHOUT_LAPACK=no -
DNOFORTRAN=0 -DDYNAMIC_ARCH=ON -DCMAKE_BUILD_TYPE=Release -
DBUILD_SHARED_LIBS=ON
```

If you get `CMake Error: The source directory "...." does not appear to contain CMakeLists.txt`, you are not in the correct directory.

```
cmake --build . --config Release
mkdir ..\install
cmake --install . --prefix ..\install -v
```

Now you have a folder named install in the OpenBlas folder with the files we need to use. You can change the last two commands if you would rather install the library elsewhere, however it is portable, so you can also move the whole folder to any location you want to after installing.

This folder referred to as install in the command given will be referred to as the folder where OpenBlas is installed in the next step.

Dependencies list and location:

This is the last step for making the custom library functions work. All the dependencies must be in the same folder as the Vensim executable (by default: "C:\Program Files\Vensim"). Copy the dlls into the same folder where the vendss64.exe file is.

- C:\Users\User\miniconda3\pkgs\libflang-11.0.1-h0e60522_20210131\Library\bin\flang.dll
- C:\Users\User\miniconda3\pkgs\libflang-11.0.1-h0e60522_20210131\Library\bin\flangtri.dll
- C:\Users\User\miniconda3\pkgs\libflang-11.0.1-h0e60522_20210131\Library\bin\libomp.dll
- C:\Users\User\miniconda3\pkgs\llvm-openmp-11.0.1-h2d74725_0\Library\bin\pgmath.dll

The next file is located in a subdirectory of the folder where OpenBlas is installed.

- \install\bin\openblas.dll

If you have installed Miniconda through chocolatey, the path to the 'miniconda' folder is: C:\tools\miniconda3

Lastly, copy the DLL containing the custom functions into the folder. This file can be downloaded from https://github.com/AManteola/external-functions-vensim. The file is called 'DLLVensimExternalFunctions.dll'.

Load into Vensim:

Once the DLL is installed, open Vensim, open the 'Tools->Options' menu and in the tab 'Startup' there is an option called 'External function library (x64)'. Click that option's browse button and find the dll called 'DLLVensimExternalFunctions.dll' in the same folder as the Vensim executable is located (by default: "C:\Program Files\Vensim").

Uninstallation:

Once you have OpenBlas installed and working, you can uninstall Miniconda3 and Visual Studio. However, it is recommended that you have a backup for all the dlls mentioned in the previous step, as you won't be able to recover them without reinstalling the programs in case they accidentally get deleted by reinstalling or updating Vensim, or any other possible mishap.

How to update the custom functions library:

To update the custom functions library, you just have to swap the old dll file with the new one.