

1) Installation guide: software needed to run OptiPlant tool

DTU-Department of Technology, Management and Economics

This document is intended to work as a installation guide of the necessary software to run OptiPlant tool. A copy of this document can also be found in the following link: <https://github.com/njbca/OptiPlant>.

The basic software needed to run OptiPlant tool are listed below:

Julia v1.8.5: The programming language that we are gonna use to formulate the optimization problems. For documentation, see: <https://docs.julialang.org/en/v1/>.

Visual Studio Code v1.74: An editor for writing and executing your Julia code. For download, go to the link: <https://code.visualstudio.com/>. You are welcome to work with any editor of your choice! For example, jupyter notebook. (Installation through anaconda - <https://sparkbyexamples.com/python/install-anaconda-jupyter-notebook/>)

JuMP v1.6.0: A package embedded in the Julia programming language. It allows users to write optimization problems. For documentation, see: <https://jump.dev/JuMP.jl/stable/>

Gurobi 10.0.0: A commercial solver for optimization problems. An academic license is available for DTU students to use. For documentation, see: <https://www.gurobi.com/documentation/>

Others: The installation guide of other necessary Julia packages to read and clean data, visualize and plot results, etc. is also provided.

All the items listed above work together as follows: An optimization problem is written in the **Julia** programming language, using the **JuMP** package syntax and **VS Code** as a text editor. Various data is imported from CSV or excel files using specific packages. Then, all the info it is passed to the solver **Gurobi**, which finds an optimal solution to the problem by using a variety of optimization approaches and techniques. Finally, the obtained results can be exported as CSV files or plotted in graphs using other specific packages.

DTU



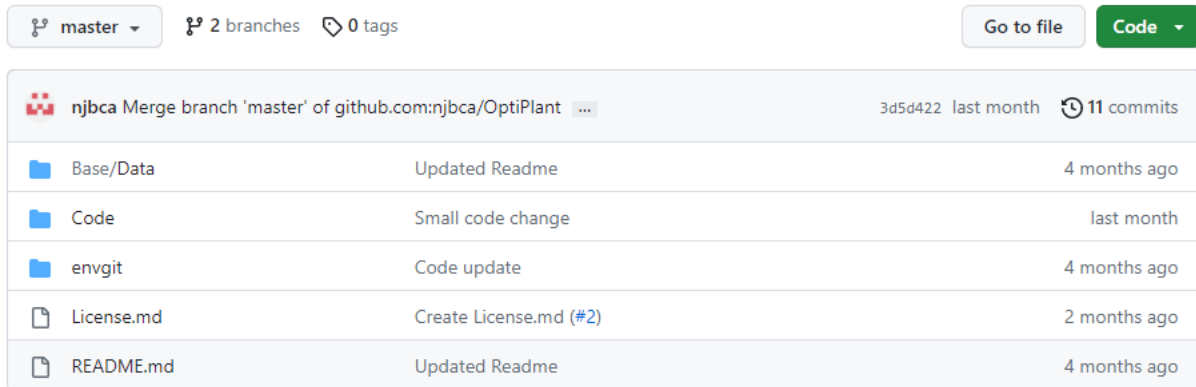
DTU Management
Department of Technology,
Management and Economics

Table of contents

GitHub: njbca/OptiPlant main page	2
'Julia' installation	3
<i>Installation steps</i>	3
'Visual Studio Code (VSCode)' installation	5
<i>Installation steps</i>	5
<i>VSCode - 'Julia' extension installation and activation of the environment</i>	8
Packages installation	11
<i>'Gurobi' licensing and installation steps</i>	11
<i>Installation steps for other packages and extensions (JuMP, CSV, ExcelReaders, Plots...)</i>	13
Final note and troubleshooting	14

GitHub: njbca/OptiPlant main page

If you go to the website <https://github.com/njbca/OptiPlant>, the front-page would look like this:



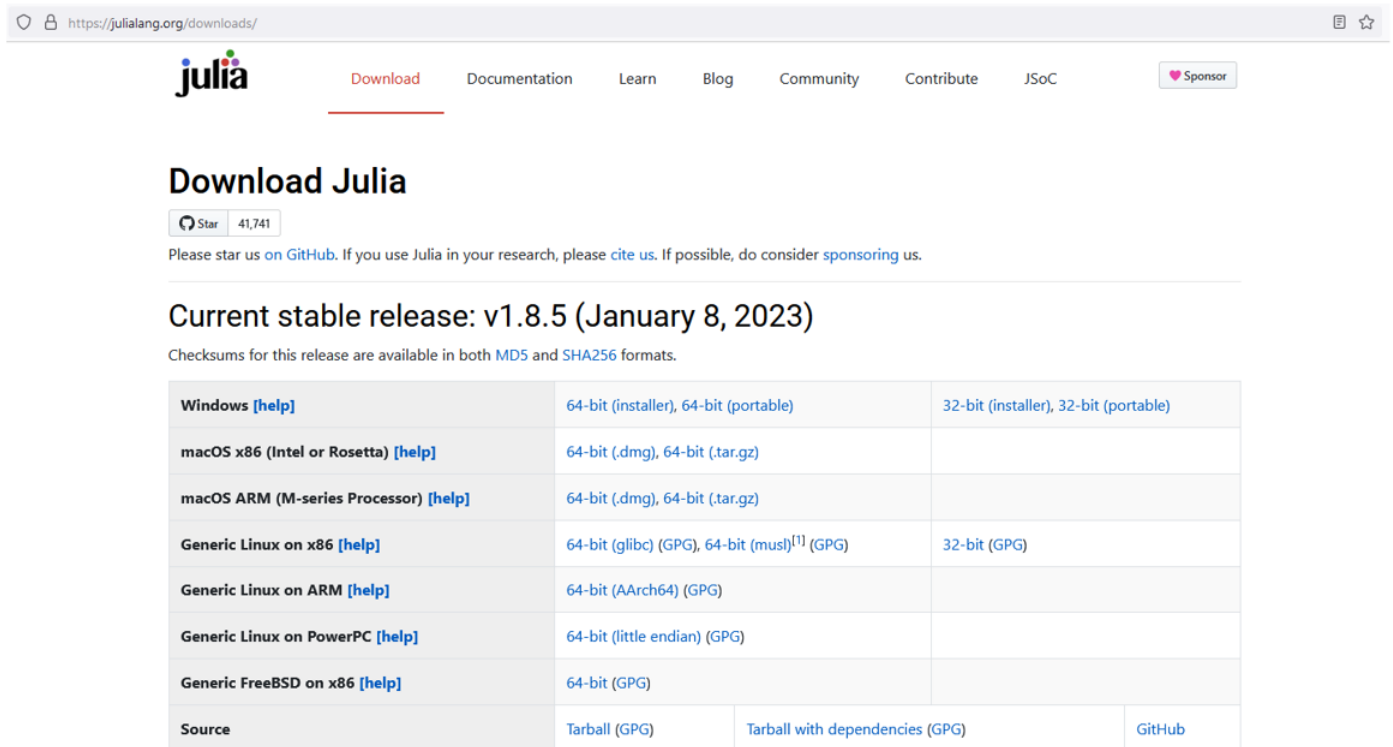
Reading through the *README.md* file displayed in the GitHub page, one will see that the first step to complete in order to run OptiPlant is to install specific software (listed in the first page of this guide). Detailed step-by-step instructions for the installation of each software is included in the following sections of this document.

'Julia' installation

Julia is a high-level, general-purpose, dynamic programming language. Its features are well suited for numerical analysis and computational science. Julia is the language that we are going to use to write our OR problems.

Installation steps

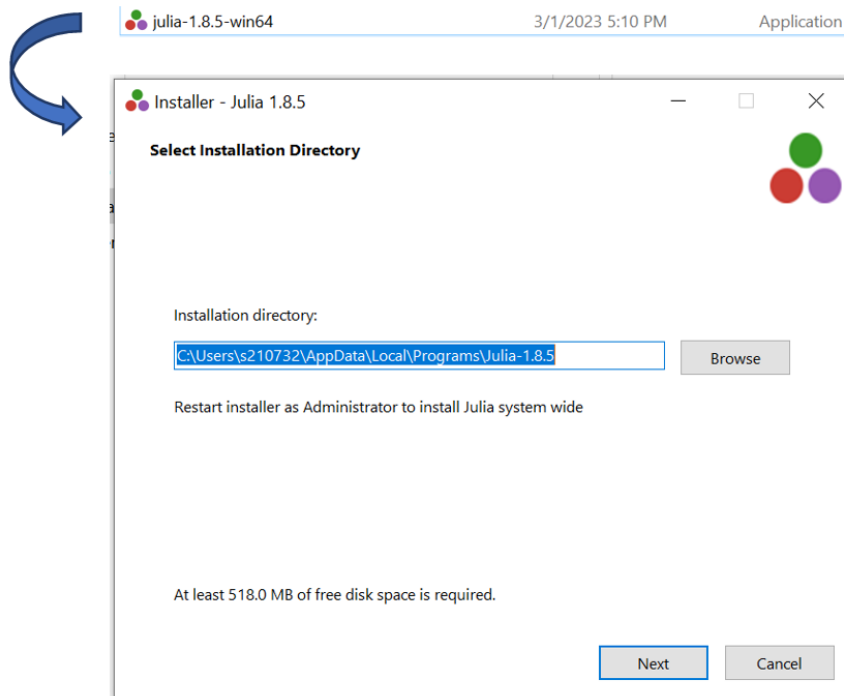
1) Go to the website: <https://julialang.org/downloads/> and download the Julia version suiting your operating system:



The screenshot shows the Julia download page. The browser address bar displays <https://julialang.org/downloads/>. The page features the Julia logo and a navigation menu with links: Download, Documentation, Learn, Blog, Community, Contribute, JSoC, and a Sponsor button. Below the navigation, the heading 'Download Julia' is followed by a GitHub Star button showing 41,741 stars. A message encourages users to star the project on GitHub, cite it, or sponsor it. The current stable release is v1.8.5 (January 8, 2023). A note states that checksums are available in MD5 and SHA256 formats. A table lists download links for various operating systems and architectures.

Windows [help]	64-bit (installer), 64-bit (portable)	32-bit (installer), 32-bit (portable)
macOS x86 (Intel or Rosetta) [help]	64-bit (.dmg), 64-bit (.tar.gz)	
macOS ARM (M-series Processor) [help]	64-bit (.dmg), 64-bit (.tar.gz)	
Generic Linux on x86 [help]	64-bit (glibc) (GPG), 64-bit (musl) ^[1] (GPG)	32-bit (GPG)
Generic Linux on ARM [help]	64-bit (AArch64) (GPG)	
Generic Linux on PowerPC [help]	64-bit (little endian) (GPG)	
Generic FreeBSD on x86 [help]	64-bit (GPG)	
Source	Tarball (GPG)	Tarball with dependencies (GPG)
		GitHub

2) Run the Julia installer and install the program:



If the installation is successful, this will appear:



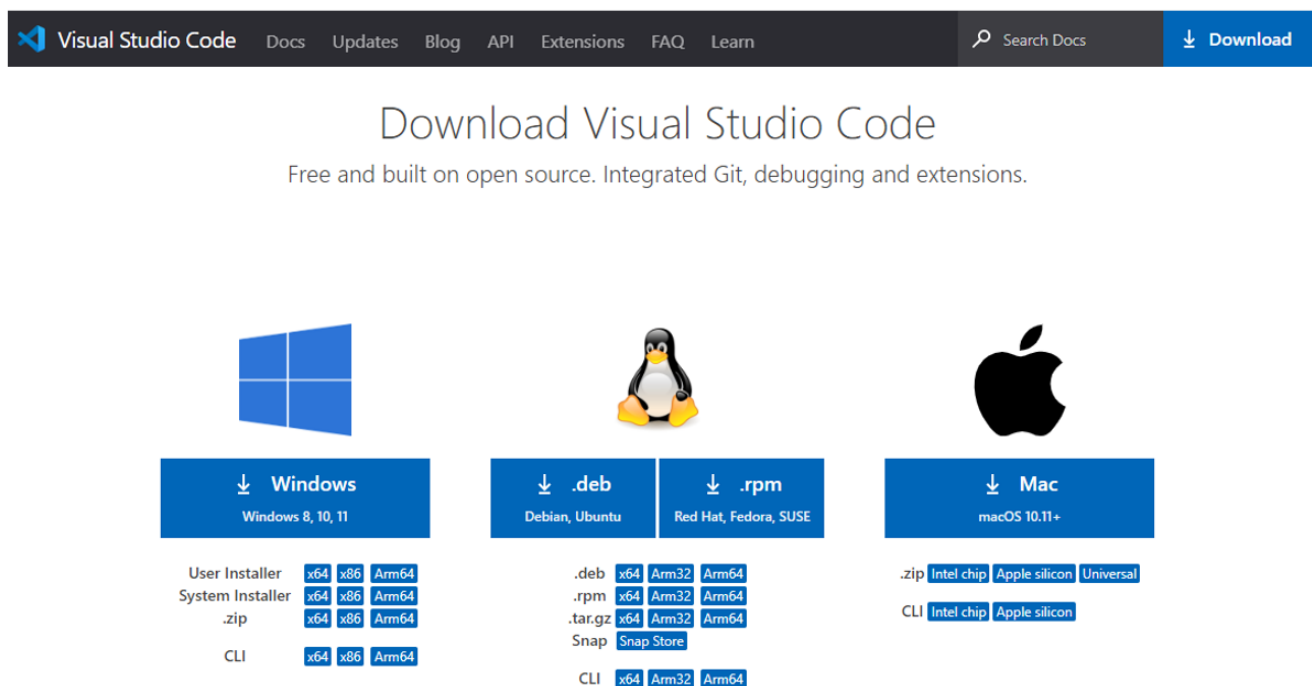
Congrats, first step completed! You just got Julia on your PC :)

'Visual Studio Code (VSCode)' installation

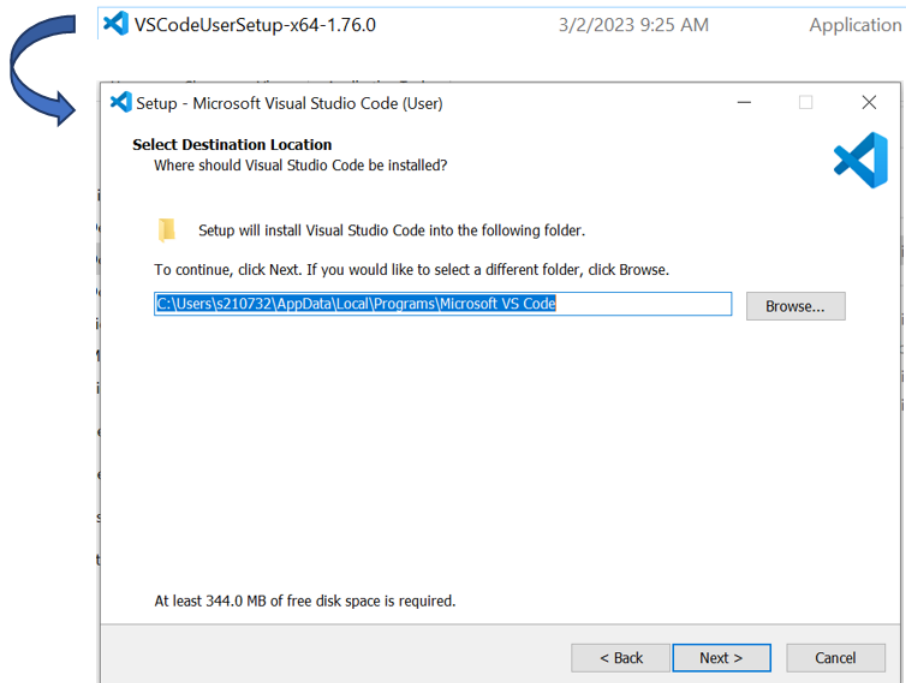
Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux. Its features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git, among others. VSCode is the editor we are going to use to write and run the Julia code.

Installation steps

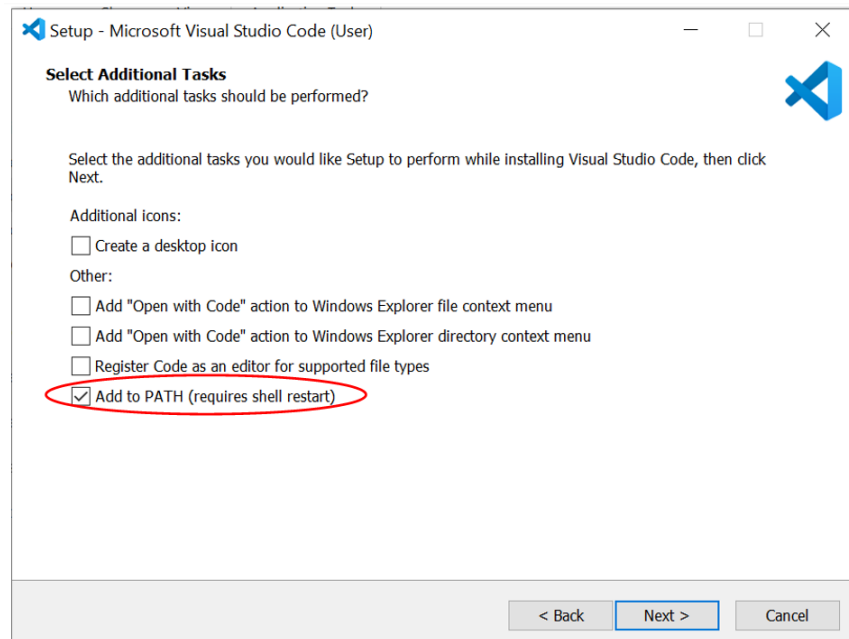
1) Go to the website: <https://code.visualstudio.com/Download> and download the version corresponding to your operating system.



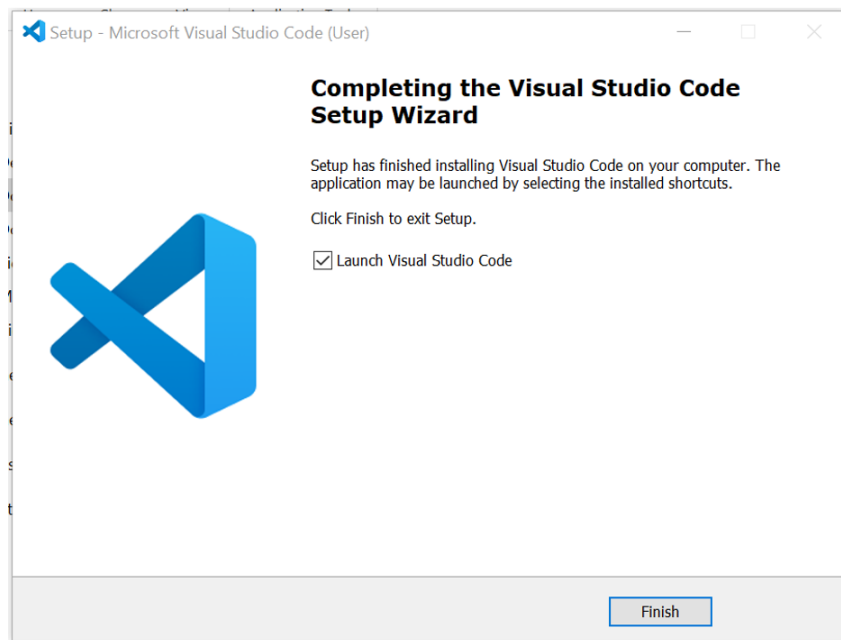
2) Run the Visual Studio Code installer and install the program:



IMPORTANT NOTE! During the installation process, remember to tick the option 'Add to PATH (requires shell restart)' -see Fig. below-:



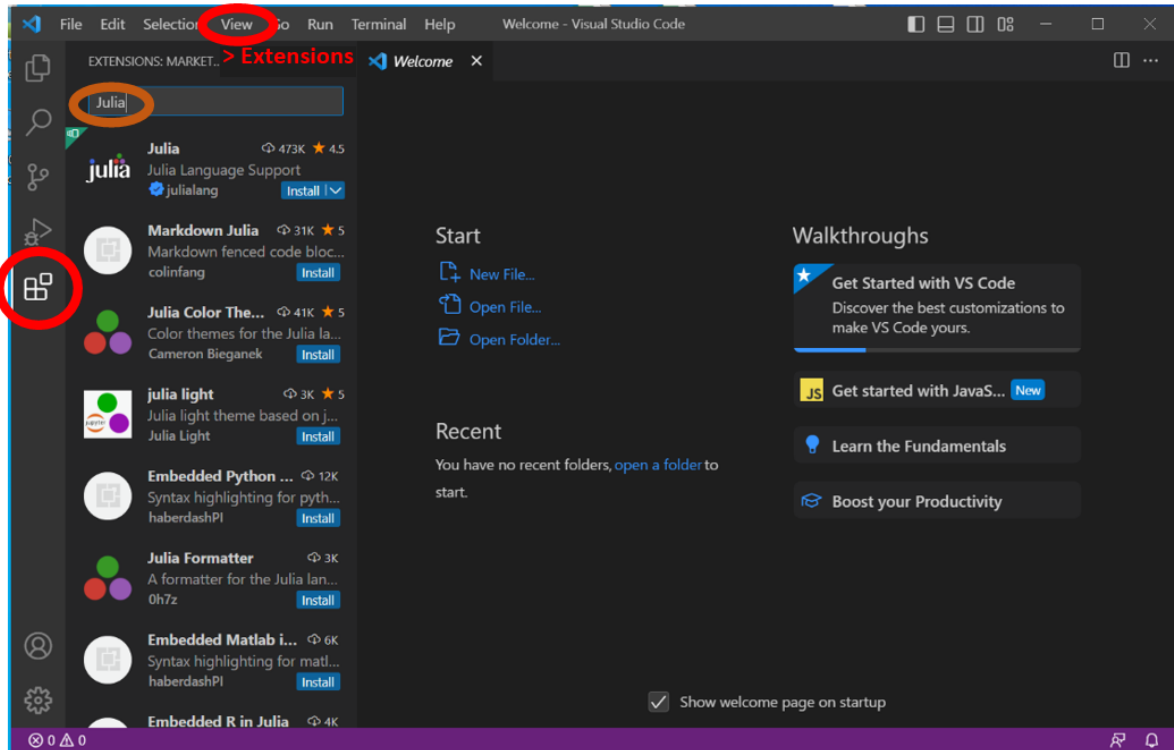
If the installation is successful, this will appear:



VSCode - 'Julia' extension installation and activation of the environment

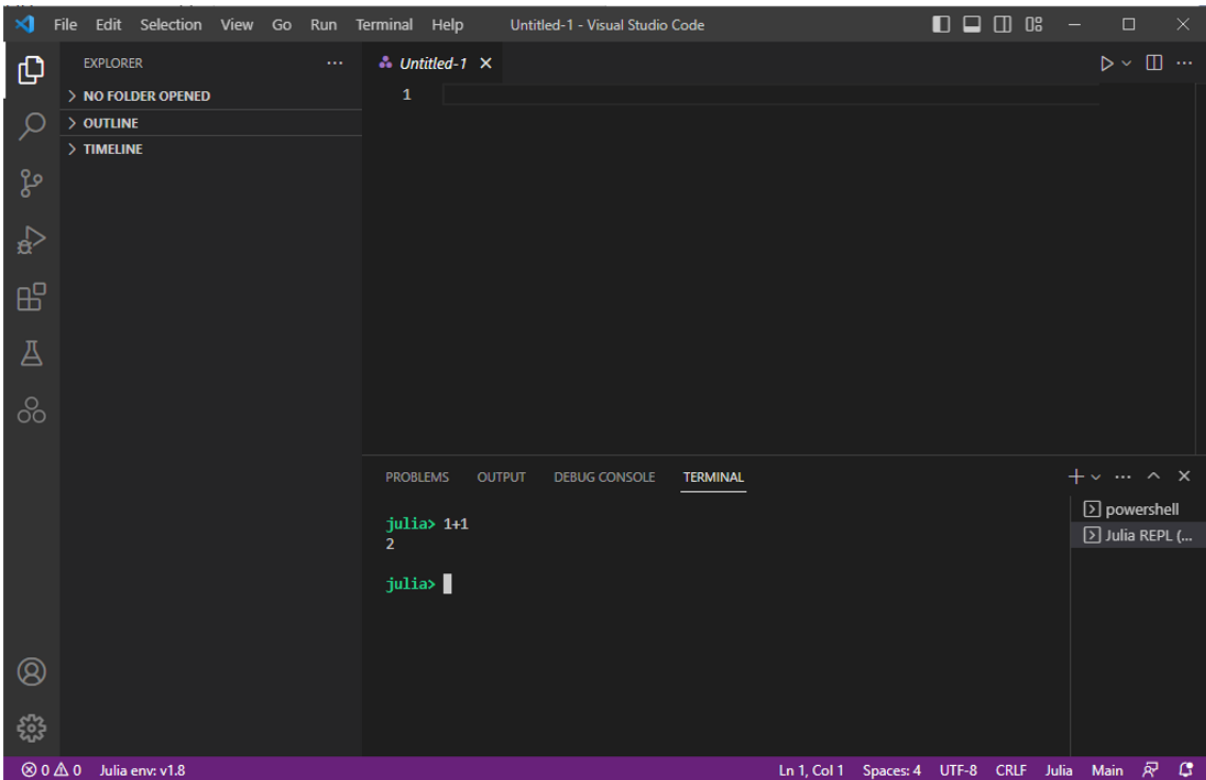
In order to be able to edit and run Julia code in VS Code, an extension has to be installed. The installed extension provides support for the Julia programming language.

To install it, open VS Code and go to 'View > Extensions' or to the fifth icon on the left. Type 'Julia' and install the extension:



For more information such as a 'start-up' guide, go to: <https://code.visualstudio.com/docs/languages/julia>

This installed extension provides a Julia REPL inside VS Code. You can start this REPL with the "Julia: Start REPL" command. This is done by pressing 'Ctrl+Shift+P' (it opens the Command Palette) and typing the command: 'Start Julia'. After this, you can start coding with Julia (e.g. $1 + 1 = 2$) -see image below-.



To enter the ‘package manager’ press ‘]’ (you will see that the colored text ‘julia>’ text changes to ‘(@v1.8) pkg>’).

The package manager lets you install, update and remove packages. You can check the installed packages by inserting the command ‘status’. The package manager also helps you set up Julia environment. (*You can go back by just pressing the backspace button*).

Activate the environment by writing ‘activate env’ on the package manager (you can see it is activated as the name ‘(@v1.8) pkg>’ will change to env). This creates a new environment/folder called ‘env’ in the specified folder:

```
(@v1.8) pkg> activate env
Activating project at `C:\Users\s210732\env`
```

You can later check whether the environment folder (env) exists in the corresponding directory: check with the command “pwd()”. This folder would store all the extensions and packages installed. To check which extensions and packages are installed in the environment, one can use the command ‘status’ -in this case some packages have been already installed-:

```
julia> pwd()
"C:\Users\s210732"

(env) pkg> status
Status `C:\Users\s210732\env\Project.toml`
 [336ed68f] CSV v0.10.9
 [a93c6f00] DataFrames v1.5.0
 [c04bee98] ExcelReaders v0.11.0
 [2e9cd046] Gurobi v1.0.0
 [4076af6c] JuMP v1.9.0
 [7a882280] LaTeXTables v0.1.1
 [23fbe1c1] Latexify v0.15.18
 [91a5bcdd] Plots v1.38.8
 [fdbf4ff8] XLSX v0.9.0
```

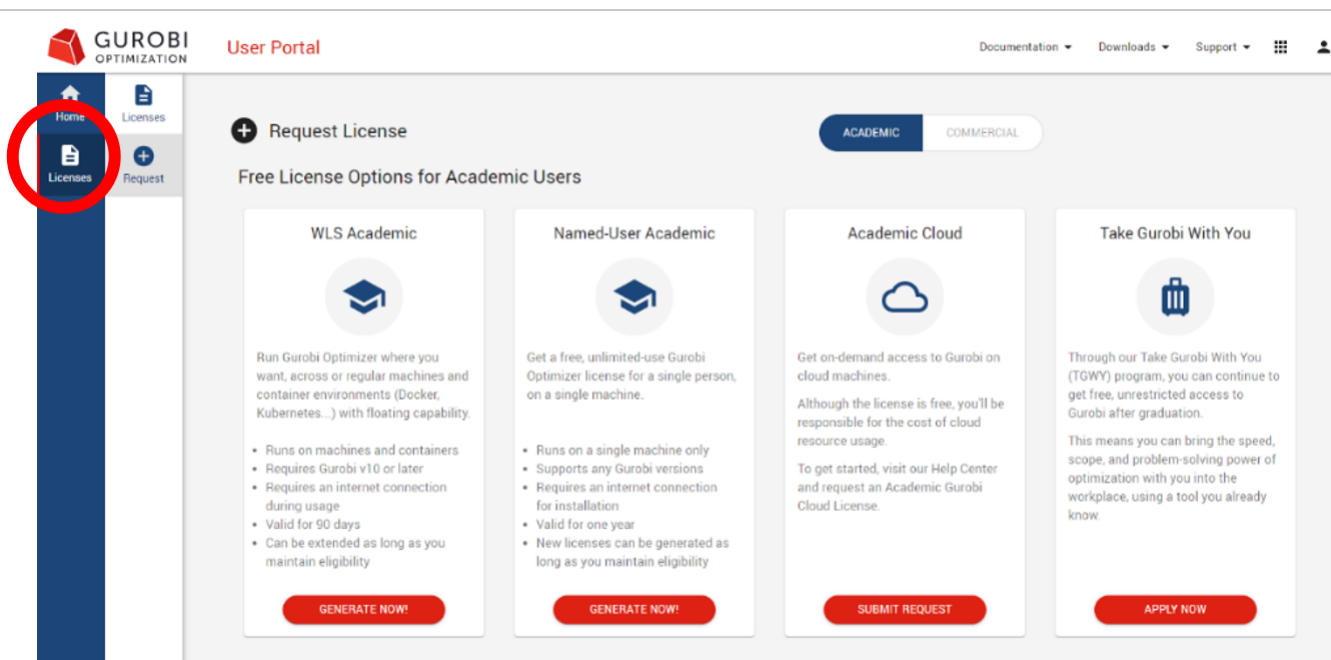
Packages installation

'Gurobi' licensing and installation steps

Gurobi is an optimization solver for linear programming, quadratic programming, quadratically constrained programming, mixed integer linear programming, mixed-integer quadratic, etc.

It can be installed from the website <https://www.gurobi.com/downloads/end-user-license-agreement-academic/> and it is possible to obtain a free academic license by registering with your DTU credentials.

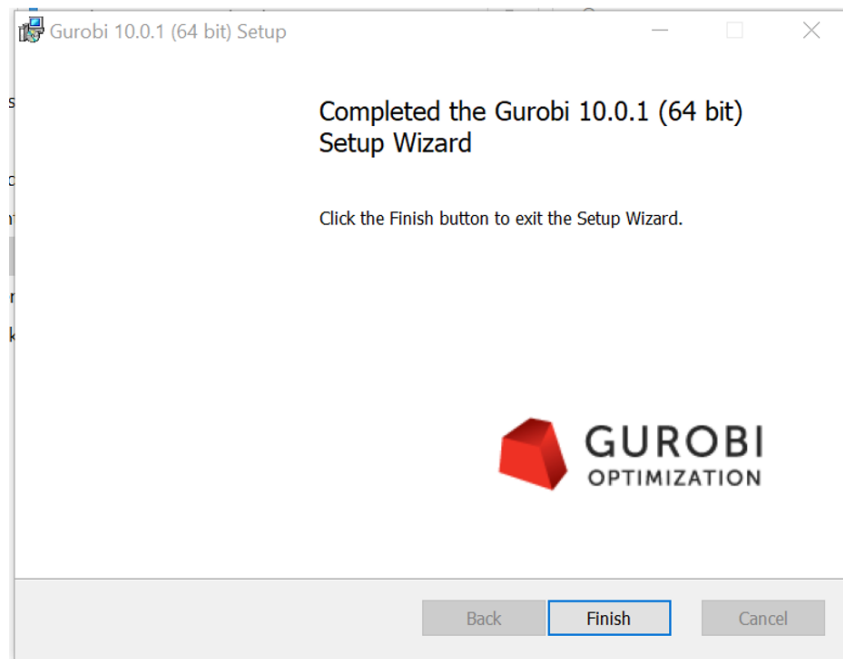
Once you are registered, go to Licences > Request, and get the license that better suits you:



IMPORTANT NOTE!: After generating the license, you will get a **grbgetkey**, it is important you copy this safely, as it is crucial during the installation of Gurobi.

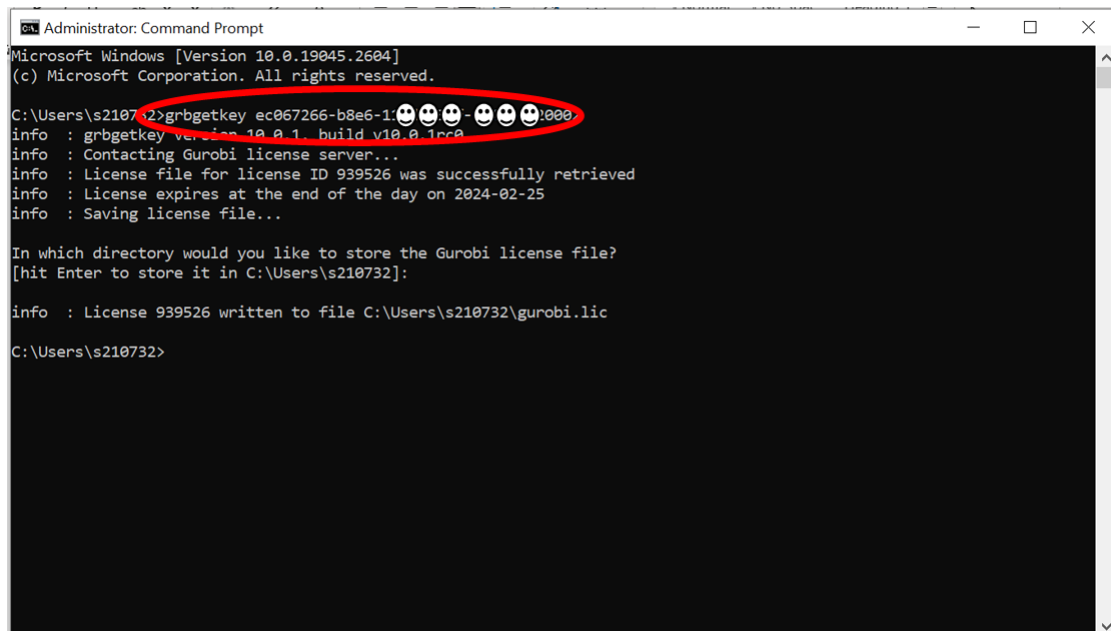
Afterwards, go to <https://www.gurobi.com/downloads/gurobi-optimizer-eula/> and install the latest version of Gurobi optimizer.

When the installation is successfully completed, this will appear:



Note: If Gurobi doesn't ask to restart your system automatically, do it manually after its installation.

Finally, open the 'Command Prompt' of your system -write "cmd" in the search menu of our PC- and insert the **grbgetkey** you have copied before ('grbgetkey *n*u*m*b*e*r'), as shown below:



Save the license in the default location suggested by the 'cmd'.

Open VS Code and start Julia. Add or Update the package “Gurobi” by writing: ‘add Gurobi’ in the ‘env’ option. As previously mentioned, the package will be saved inside the ‘env’ folder:

```
(env) pkg> add Gurobi
Updating registry at `C:\Users\s210732\.julia\registries\General.toml`
Resolving package versions...
No Changes to `C:\Users\s210732\env\Project.toml`
No Changes to `C:\Users\s210732\env\Manifest.toml`
```

You are done with Gurobi installation!

Installation steps for other packages and extensions (JuMP, CSV, ExcelReaders, Plots...)

Some other packages such as ‘JuMP’, ‘CSV’, ‘ExcelReaders’, or ‘Plots’ would also be needed to perform a simulation using the OptiPlant tool and to display the results in graphs.

The installation of these packages is quite easy: After activating the environment (env), write the following command: ‘add ***’ (***= name of the package). The installation of the packages will start automatically and they will be saved inside the ‘env’ folder.

Remember that you can check all the installed packages in your environment by writing the command ‘status’ inside the ‘env’ dialogue -see an example below-:

```
(env) pkg> status
Status `C:\Users\s210732\env\Project.toml`
[336ed68f] CSV v0.10.9
[a93c6f00] DataFrames v1.5.0
[c04bee98] ExcelReaders v0.11.0
[2e9cd046] Gurobi v1.0.0
[4076af6c] JuMP v1.9.0
[7a882280] LaTeXTables v0.1.1
[23fbe1c1] Latexify v0.15.18
[91a5bcd] Plots v1.38.8
[fdbf4ff8] XLSX v0.9.0
```

Final note and troubleshooting

Remember to activate the environment every time you start VSCode in order to have the different packages called.

As mentioned before, to do so you have to enter the ‘package manager’ by pressing ‘]’ -you will see that the colored text ‘julia>’ text changes to ‘(@v1.8) pkg>’. Next, activate the environment by writing ‘activate env’ on the package manager -you can see it is activated as the name ‘(@v1.8) pkg>’ will change to ‘env’.

For any troubles on the installation of the different software hereby mentioned, check the official installation guides for each program.

I hope this installation guide guide was useful! :)

*After completing all the steps listed on this guide, one can download all the elements of the OptiPlant tool found in the same GitHub page (<https://github.com/njbca/OptiPlant>). There, one can also find another manual called **’2) Optiplant tool: user guide’** that describes the different components and elements that comprise the tool and how to properly use it.*