

How to install RMS under Windows

This document was written by Peter Eschman, and is based on a recorded TeamViewer session on his computer, under the direction of Denis Vida.

Start by downloading these three resources:

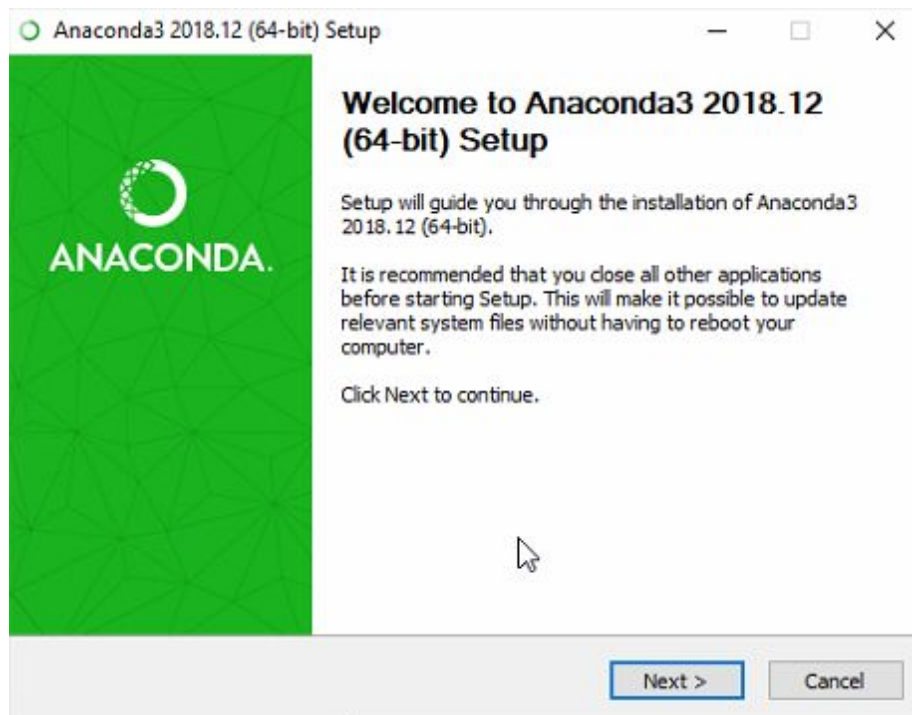
1) <http://go.microsoft.com/fwlink/?Linkid=691126>
download Visual C++ 2015 x86 x64 Cross Build Tools
Install the tools

2) <https://www.anaconda.com/download>
download the install for Python 3.7

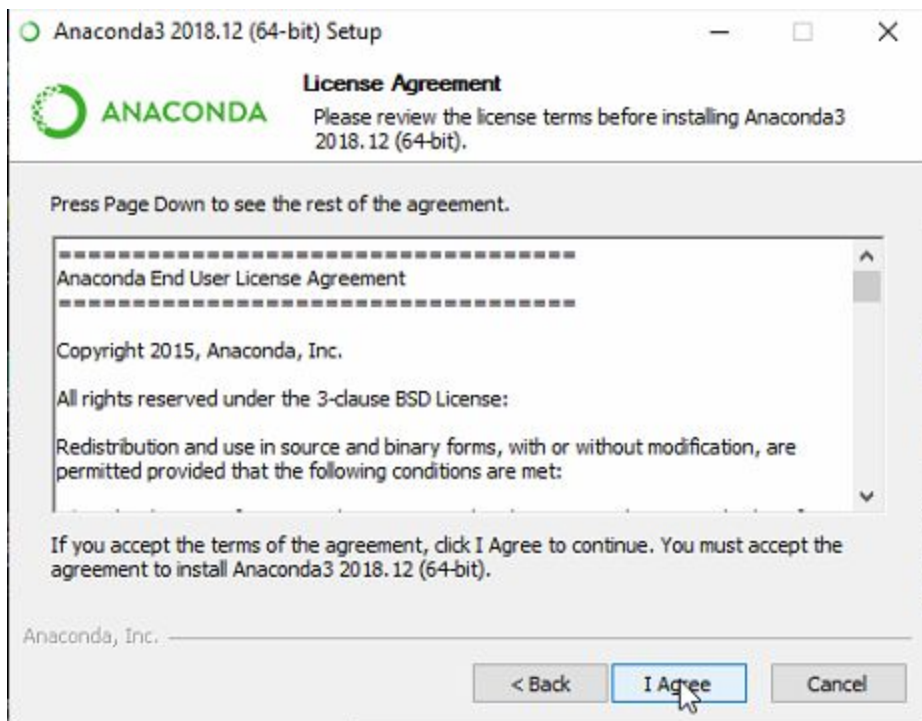
3) <https://git-scm.com>
download Git

Continuing step 2), installing Anaconda:

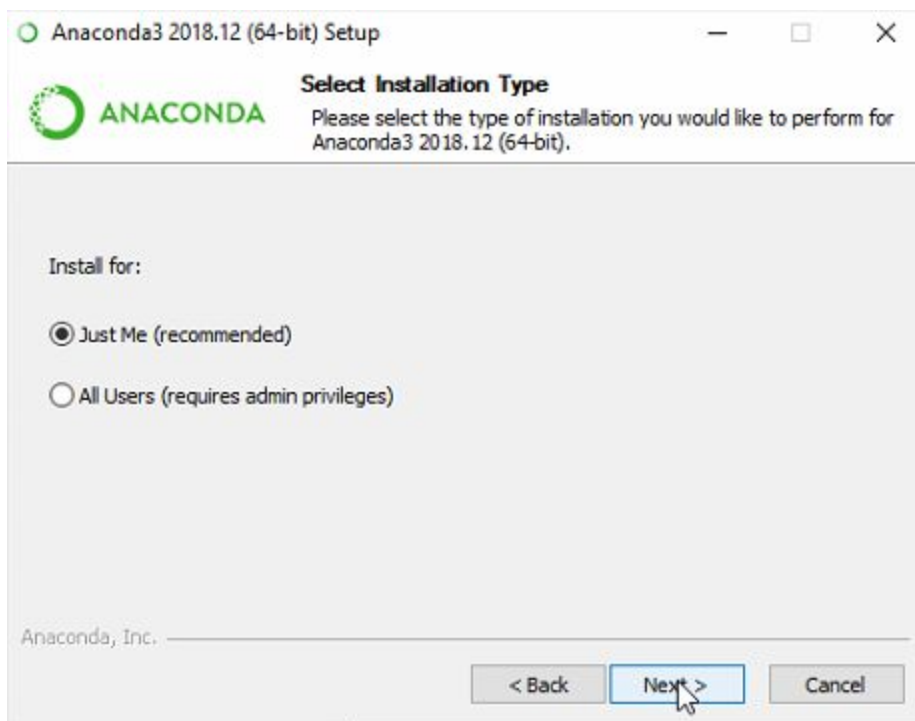
First you will see the Welcome screen, so click Next:



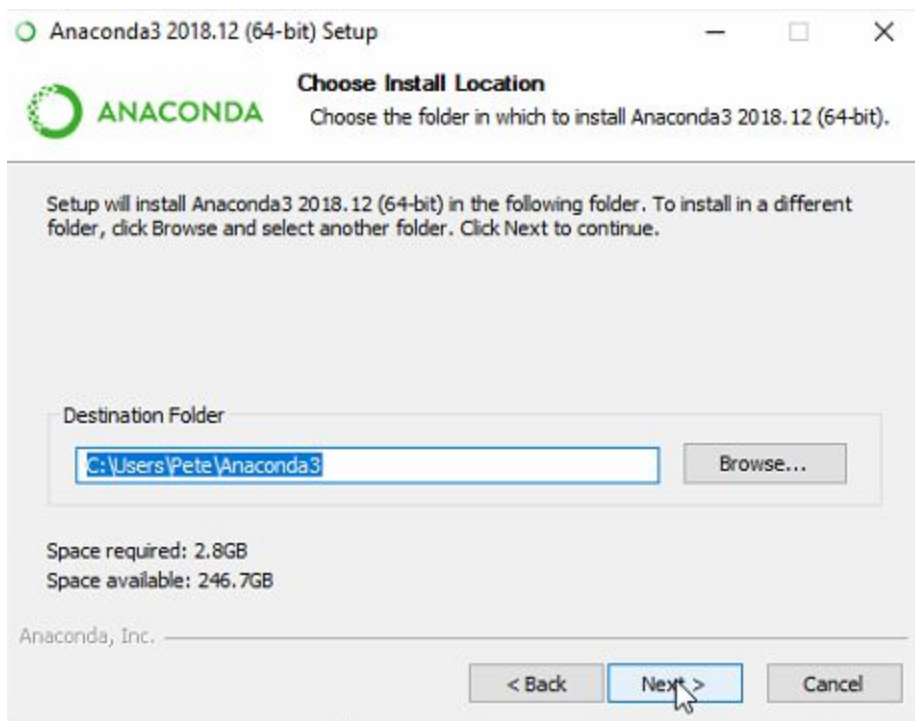
Click I Agree to accept license:



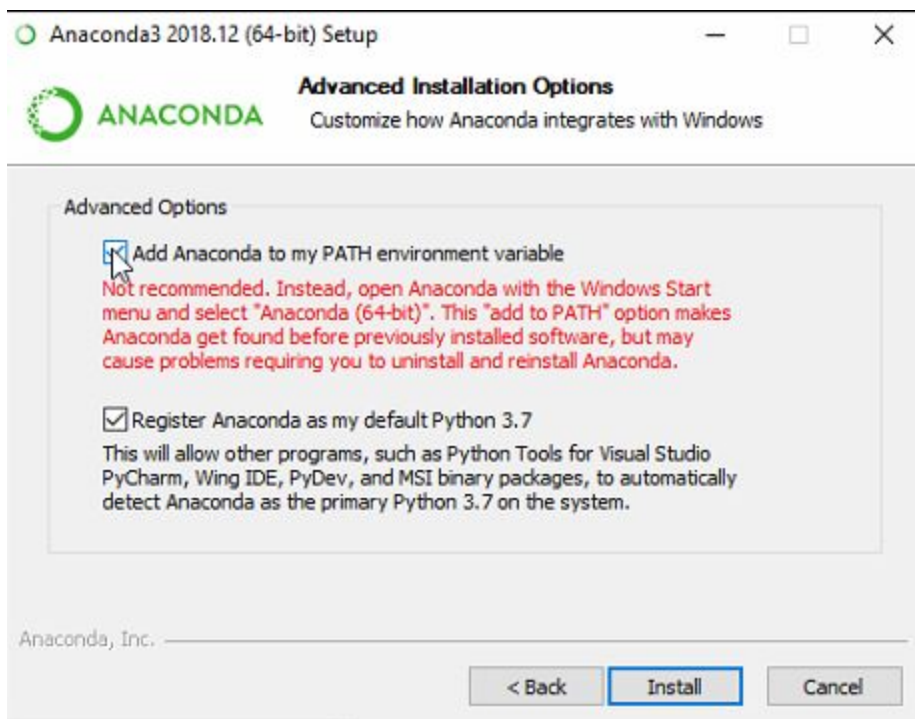
You need to install for "just me" on the next setup screen, otherwise the install will not work correctly:



Choose destination folder next, the default location is fine:

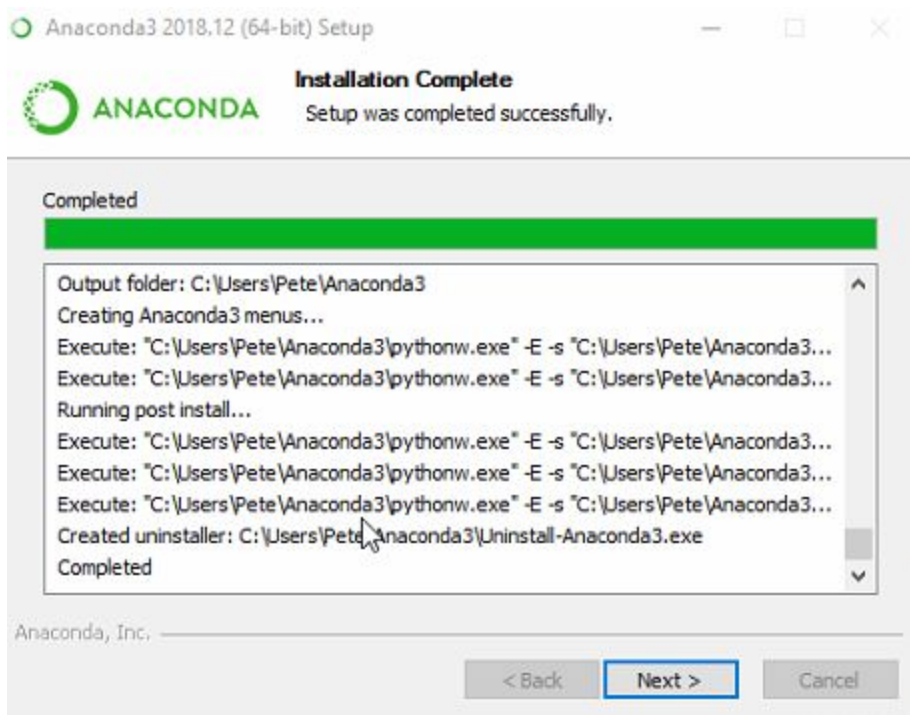


On the next screen, it is best to select the option to add Anaconda to the PATH environment variable,, then click Install. If you don't do this, you will not be able to call conda from Windows PowerShell, so do this unless you are using another active installation of Python:

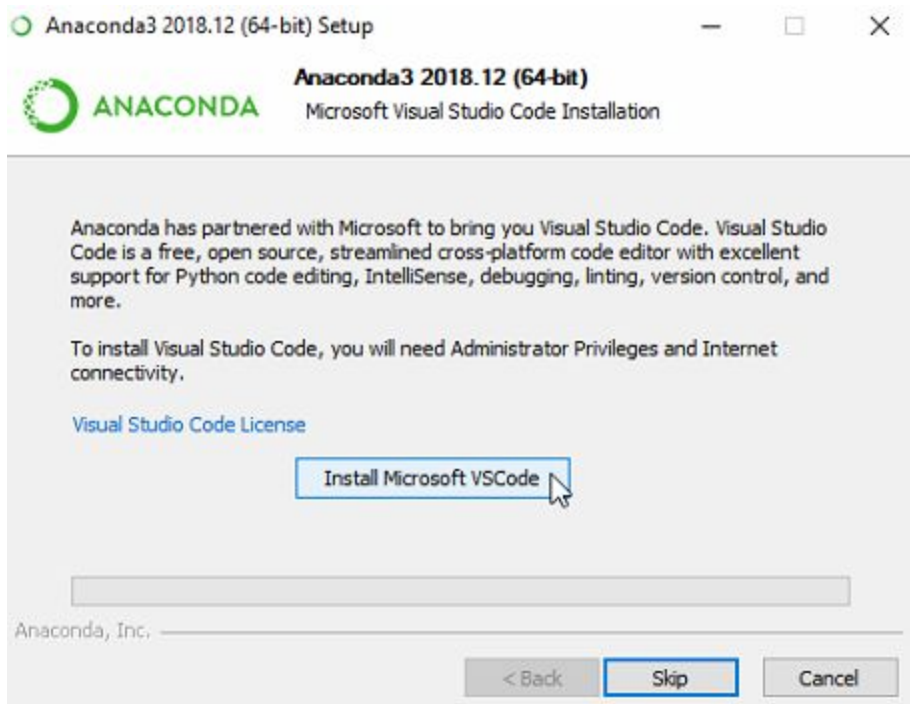


After you have clicked Install, it may take a while for the install to complete. On an older Asus N61J machine this took about 52 minutes to complete. On a newer Dell laptop, this took about 21 minutes.

When the window says "Completed", click Next



Now click the button that says Install Microsoft VSCode:

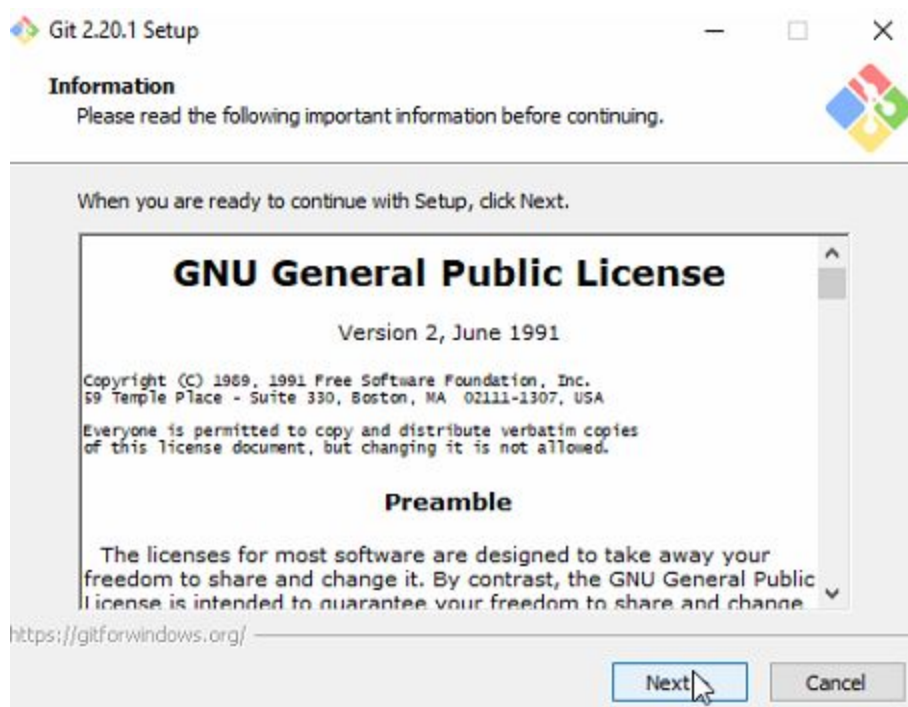


At the end of the process, it will say "Thanks for installing Anaconda" You can elect to "Learn more about Anaconda Cloud" and "Learn how to get started with Anaconda", or skip these steps, since we are primarily interested in getting RMS running, so uncheck these options and click "Finish":

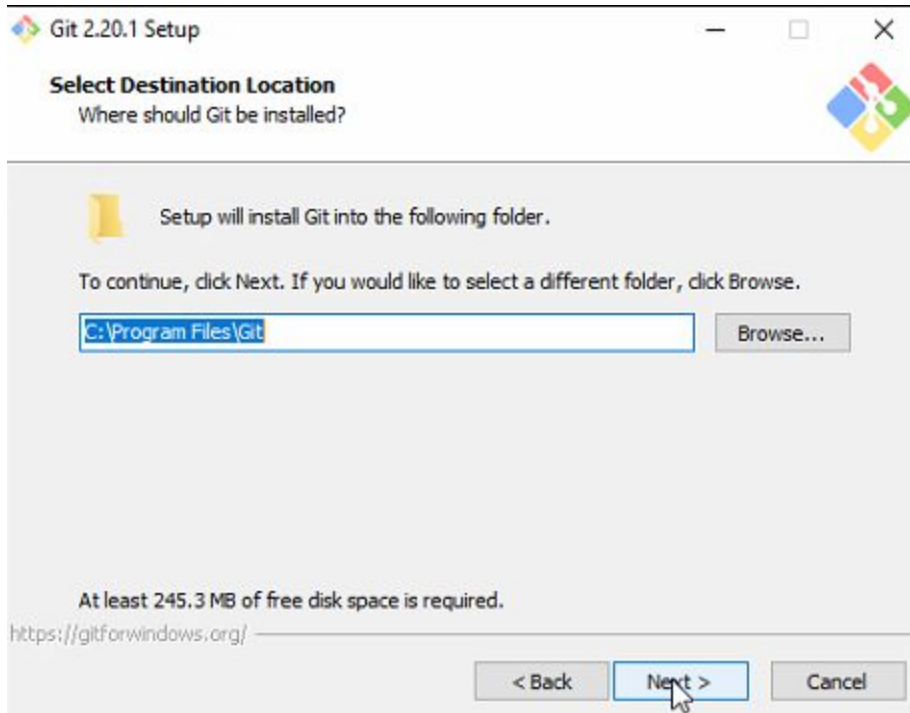


Continuing step 3) Install Git

Click "Next" on GNU license:



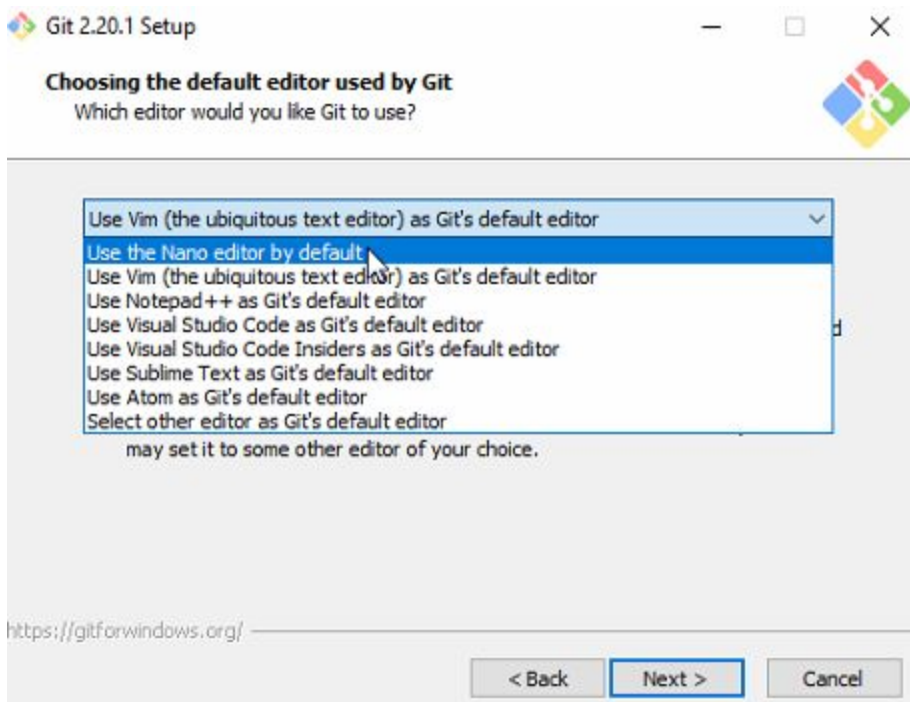
Now you can accept the default install folder:



On the next screen, you can leave the default install components alone, then click "Next".

On the next screen you can leave the Start Menu Folder on default, then click "Next".

The next step is to "Choosing the default editor used by Git". At this point you can select the option to change the editor from Vim to Nano, since this is the typical editor for the Raspberry Pi. Make your selection, then click "Next":



On the next screen: "Adjusting your PATH environment", leave settings on default and click "Next".

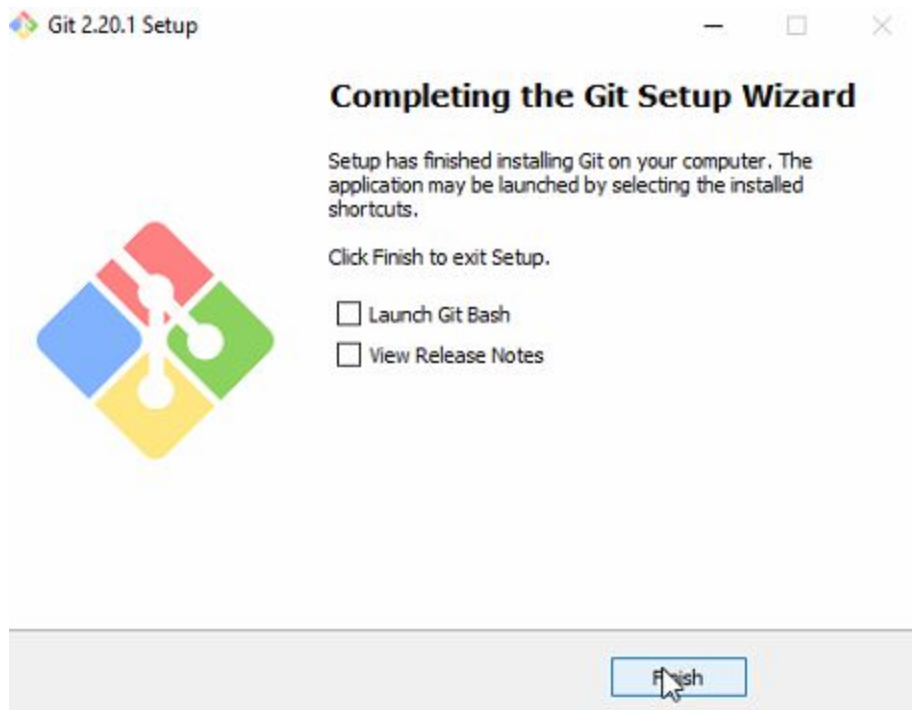
On the next screen: "Choosing HTTPS transport backend", leave settings on default and click "Next".

On the next screen: "Configuring the line ending conventions", leave settings on default and click "Next".

On the next screen: "Configuring the terminal emulator to use with Git Bash", leave settings on default and click "Next".

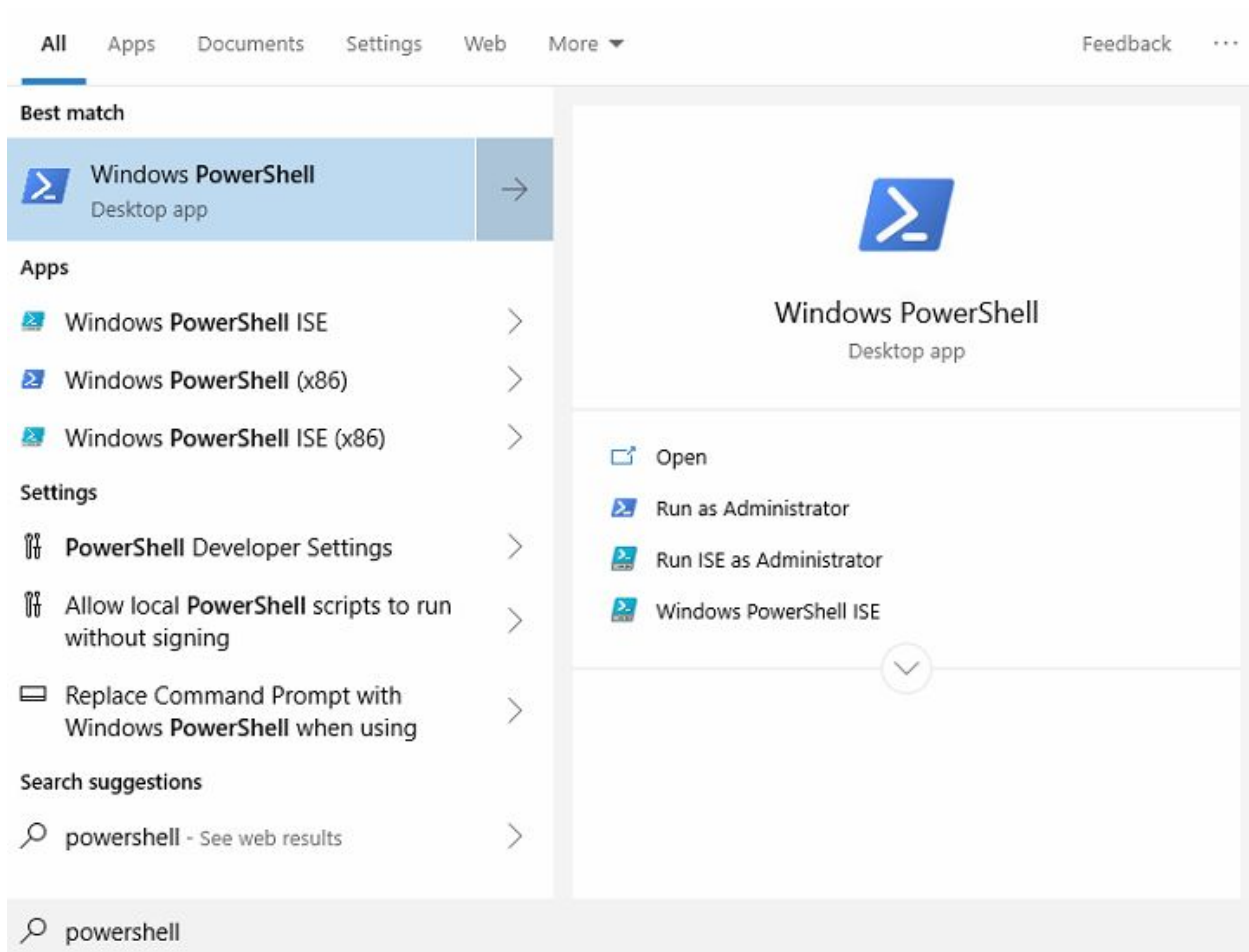
On the next screen: "Configuring extra options", leave settings on default and click "Install".

Next you will see "Completing the Git Setup Wizard", and you can elect to "View Release Notes", or just click "Finish" at this point:



Final steps, and running RMS under Windows:

After you have completed the Git install, start up Windows PowerShell (you can do this by opening the start menu, and typing "Power", then select Windows PowerShell, and open using the default settings:



The next steps are tests to see if both conda and git work from PowerShell.

type "conda" and wait to see the help options

type "git" and wait to see the help options


```
Windows PowerShell
inspect
metapackage
render
server
skeleton
verify
PS C:\Users\Pete> git
usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
      [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
      [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
      [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
      <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
  clone      Clone a repository into a new directory
  init       Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
  add        Add file contents to the index
  mv         Move or rename a file, a directory, or a symlink
  reset      Reset current HEAD to the specified state
  rm         Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)
  bisect     Use binary search to find the commit that introduced a bug
  grep       Print lines matching a pattern
  log        Show commit logs
  show       Show various types of objects
  status     Show the working tree status

grow, mark and tweak your common history
  branch     List, create, or delete branches
  checkout   Switch branches or restore working tree files
  commit     Record changes to the repository
  diff       Show changes between commits, commit and working tree, etc
  merge      Join two or more development histories together
  rebase     Reapply commits on top of another base tip
  tag        Create, list, delete or verify a tag object signed with GPG

collaborate (see also: git help workflows)
  fetch      Download objects and refs from another repository
  pull       Fetch from and integrate with another repository or a local branch
  push       Update remote refs along with associated objects

'git help -a' and 'git help -g' list available subcommands and some
concept guides. See 'git help <command>' or 'git help <concept>'
to read about a specific subcommand or concept.
PS C:\Users\Pete>
```

Next copy and paste the following three commands into PowerShell. Hit return after pasting each command (if you are prompted: "Proceed ([y]/n)? type y)

```
conda install -y numpy scipy gitpython cython matplotlib
```

```
conda install -y -c conda-forge pyephem opencv Pillow
```

```
conda install -y -c astropy astropy
```

After these completing steps, Git has installed the basic required libraries.

```
Windows PowerShell

environment location: C:\Users\Pete\Anaconda3

added / updated specs:
- opencv
- pillow
- pyephem

The following packages will be downloaded:

package | build | size | source
-----|-----|-----|-----
py-opencv-3.4.1 | py37h1b0d24d_3 | 1.5 MB | 
pyephem-3.7.6.0 | py37hfa6e2cd_1000 | 684 KB | conda-forge
libopencv-3.4.1 | h875b8b8_3 | 37.0 MB | 
opencv-3.4.1 | py37h6fd60c2_3 | 9 KB | 
conda-4.5.12 | py37_1000 | 660 KB | conda-forge
-----|-----|-----|-----
Total: | 39.8 MB |

The following NEW packages will be INSTALLED:

libopencv: 3.4.1-h875b8b8_3
opencv: 3.4.1-py37h6fd60c2_3
py-opencv: 3.4.1-py37h1b0d24d_3
pyephem: 3.7.6.0-py37hfa6e2cd_1000 conda-forge

The following packages will be UPDATED:

conda: 4.5.12-py37_0 --> 4.5.12-py37_1000 conda-forge

Proceed ([y]/n)? y

Downloading and Extracting Packages
py-opencv-3.4.1 | 1.5 MB | ##### | 100%
pyephem-3.7.6.0 | 684 KB | ##### | 100%
libopencv-3.4.1 | 37.0 MB | ##### | 100%
opencv-3.4.1 | 9 KB | ##### | 100%
conda-4.5.12 | 660 KB | ##### | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
PS C:\Users\Pete> conda install -y -c astropy astropy
Solving environment: done

# All requested packages already installed.

PS C:\Users\Pete>
```

Next, make new directory for source code, or leave on c:\users\xxx\
(I chose making a directory called d:\meteor).

Now change into the source code directory of your choice, and Clone the RMS library by copying and pasting the following command into PowerShell :

```
git clone https://github.com/CroatianMeteorNetwork/RMS.git
```

```
PS D:\> md Meteor

Directory: D:\

Mode                LastWriteTime         Length Name
----                -
d-----          12/28/2018 12:33 PM             Meteor

PS D:\> cd meteor
PS D:\meteor> git clone https://github.com/CroatianMeteorNetwork/RMS.git
Cloning into 'RMS'...
remote: Enumerating objects: 174, done.
remote: Counting objects: 100% (174/174), done.
remote: Compressing objects: 100% (107/107), done.
remote: Total 4536 (delta 100), reused 121 (delta 67), pack-reused 4362
Receiving objects: 100% (4536/4536), 42.34 MiB | 8.13 MiB/s, done.
Resolving deltas: 100% (3395/3395), done.
Checking out files: 100% (139/139), done.
PS D:\meteor> ls

Directory: D:\meteor

Mode                LastWriteTime         Length Name
----                -
d-----          12/28/2018 12:33 PM             RMS

PS D:\meteor>
```

Once this process is done, change into the new RMS directory just created in the previous step.

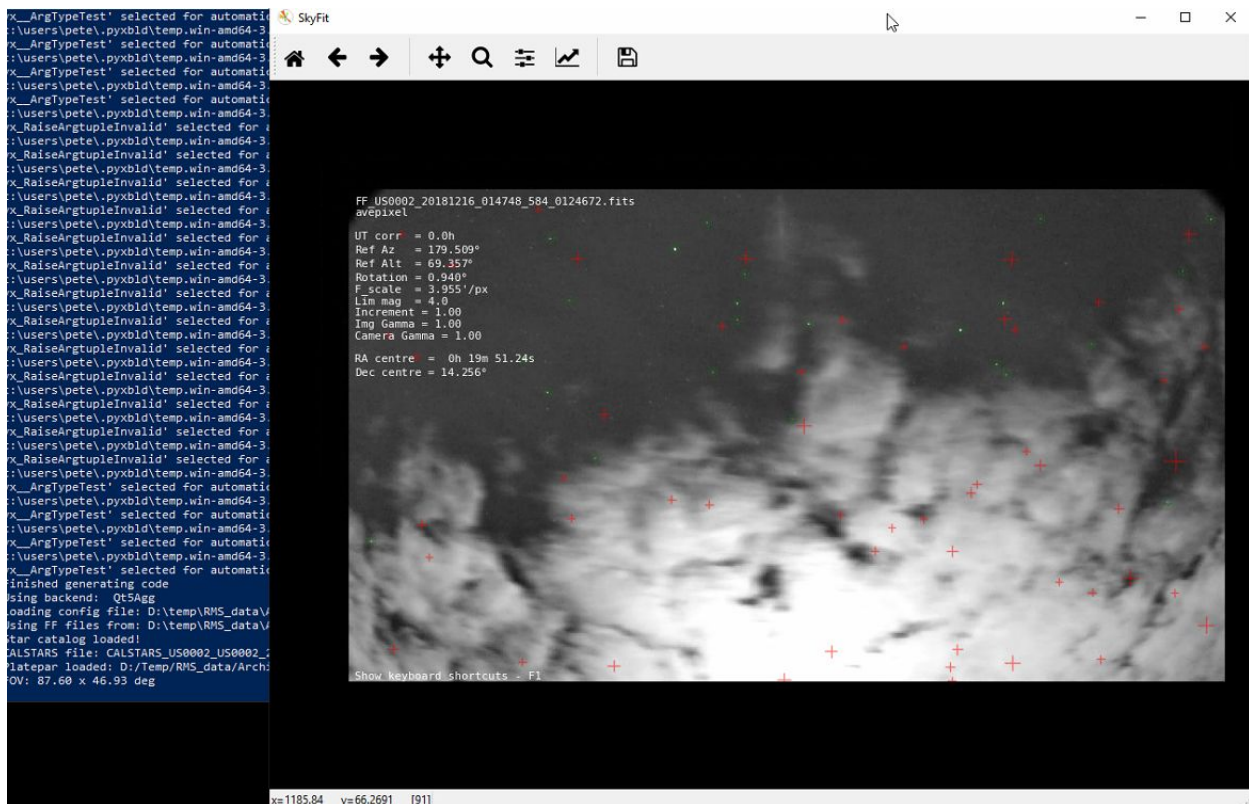
If you type: `python` , the prompt will change to `>>>`
To leave the python environment, type: `exit()`

Please note that the first time you run a python RMS command, the environment will need build for the first time. Now you can execute some typical commands, like these, assuming your data directory is something like this:

`\temp\RMS_data\ArchivedFiles\US0002_20181216_002400_040473`

You can test with commands link this one (this is all on one line, not folded):

```
python -m RMS.Astrometry.SkyFit
D:\temp\RMS_data\ArchivedFiles\US0002_20181216_002400_040473 --config .
```



When you are done, you can close PowerShell by typing `exit`

To open and use later, Start PowerShell, change into your new source directory:
`cd meteor\rms`
 and run some typical commands, like stacking:

```
python -m Utils.StackFFs -s
\\temp\MCam\RMS_data\ArchivedFiles\US0002_20181206_002203_521407.png
```

Or

```
python -m Utils.StackFFs -s d:\temp\RMS_data\BobH_07\1214_07.png
```

Please note that RMS commands may require that your data is located under a directory named "RMS_data".

I hope you will enjoy your new and powerful RMS command environment, I know I will.