USER GUIDE

Alexander Vevstad, Amanda Aasa, Amanda Svennblad,
Daniel Thomas, Lina Larsson, Olav Berg

Version 1.0

Parts of chapter 1 and 2 are based on the 2019 user guide for ENVISIoN.

Status

Granskad		
Godkänd		

PROJECT IDENTITY

2020/Spring semester,
Faculty of Science and Engineering, Linköping University, IFM

Group members

Name	**Role**	**Phone nr.**	**E-mail**
Alexander Vevstad	Project leader (PL)	079-3337906	aleve534@student.liu.se
Amanda Aasa	Document manager(DOK)	070-3618520	amaaa939@student.liu.se
Amanda Svennblad	Amanda Svennblad (AS)	073-5410046	amasv271@student.liu.se
Daniel Thomas	Daniel Thomas (DT)	073-6438052	danth168@student.liu.se
Lina Larsson	Lina Larsson (LL)	070-6781016	linla245@student.liu.se
Olav Berg	Olav Berg (OB)	070-4833500	olabe123@student.liu.se

Website: https://liuonline.sharepoint.com/sites/Lisam_TFYA75_2020VT_GK/67869

Client: Rickard Armiento, IFM, Linköping University, 581 83 Linköping
Contact person of client: Rickard Armiento, rickard.armiento@liu.se
Course examintor: Per Sandström, per.o.sandstrom@liu.se
Main supervisor: Joel Davidsson, joel.davidsson@liu.se

Contents

Document history

Version	Date	Changes	Done by	Reviewed
0.1	2020-05-22	First draft.	AA, AS, OB, DT, LL, AV	LL
0.2	2020-05-25	Small fixes such as spelling	AS, LL	AA
0.2	2020-06-01	Updated version	LL	AA

Licens

This documet is licensed as BSD, see appendix A.

1 Introduction

ENVISIoN is an open source toolkit for electron visualization, developed as a part of the course TFYA75: Applied Physics - Bachelor's Project, given at Linköping University, LiU. It's implemented by using a modified verision of the Inviwo visualization framework, developed at the Scientific Visualization Group at Linköping University, LiU.

The present version was developed during the spring semester of 2020 by a project group consisting of: Alexander Vevstad, Amanda Aasa, Amanda Svennbland, Daniel Thomas, Lina Larsson and Olav Berg. The Supervisor was Joel Davidsson, the Requisitioner and co-supervisor was Rickard Armiento and the Visualisation expert was Peter Steneteg. The Course examiner was Per Sandström. The work is based on a previous version by the project group taking the course in the spring semester of 2019 consisting of: Linda Le, Abdullatif Ismail, Anton Hjert, Lloyd Kizito and Jesper Ericsson, with the Supervisor Johan Jönsson, the Requisitioner and co-supervisor Rickard Armiento, the Visualisation expert Peter Steneteg and the Course examiner Per Sandström. That work was based on a previous version by the project group taking the course in the spring semester of 2018 consisting of: Anders Rehult, Marian Brännvall, Andreas Kempe and Viktor Bernholtz with the Supervisor Johan Jönsson, the Requisitioner and co-supervisor Rickard Armiento, the Visualisation expert Rickard Englund and Course examiner Per Sandström. That work was is based on the work by the project group taking the course in the spring term of 2017 consisting of: Josef Adamsson, Robert Cranston, David Hartman, Denise Härnström and Fredrik Segerhammar. Supervisor: Johan Jönsson; Requisitioner and co-supervisor: Rickard Armiento; Visualization expert: Peter Steneteg; and Course examiner: Per Sandström.

ENVISIoN provides a graphical user interface and a set of Python scripts that allow the user to:

- Read and parse output from electronic structure calculations made by the program VASP and storing the result in a structured HDF5 file.
- Generate Inviwo visualizations for common tasks when analyzing electronic structure calculations. Presently there is support for visualizing the crystal structure of the unit cell of a material, electron localization function (ELF)-data, electronic charge density, electronic band structure, radial Distribution Function and density of states - both total and partial. The system also provides the ability to interconnect some of the networks mentioned above.

2 How to build Inviwo with ENVISIoN on Ubuntu 18.04 LTS

These instructions show how to build Inviwo and ENVISIoN on Ubuntu 18.04 LTS.

2.1 Install git

Start by installing git, which will be used to fetch ENVISIoN in the next step.

```
sudo apt install git
```

2.2 Download ENVISIoN

Go to your home folder and clone ENVISIoN from Github. This guide will assume that both ENVISIoN and Inviwo will be placed directly under the home folder.

```
cd
git clone https://github.com/rartino/ENVISIoN
```

2.3 Prepare Inviwo using the ENVISIoN install script

ENVISIoN provides an install script for Ubuntu 18.04 LTS. Executing the installation script will install all required dependencies, clone Inviwo from Github and configure the Inviwo build.

The script should *NOT* be run as root, but as your own user and it will ask for your password when it needs root rights. It is possible that the script will ask for other user input during the process, if that's the case, just accept the default.

```
cd ~/ENVISIoN/scripts
./install.sh /home/$USER/ENVISIoN /home/$USER/inviwo
```

Once the installation script has run, it prints build instructions. Follow the instructions and start the build. The instructions will tell you to *cd* to the build directory and execute make.

An easy way to modify the build settings, if needed, is to install the cmake curses gui and run it in the build directory.

To install the cmake gui:

```
sudo apt install cmake-curses-gui
```

Running cmake in the build directory:

```
cd ~/inviwo/build
ccmake .
```

When in the GUI, press "Configure" to apply the current configuration, "Generate" to generate build files and *q* to quit. If settings have changed, it is possible that you will need to press "Configure" more than once before the "Genereate" option becomes available.

After having generated the build files, the project can now be rebuilt with the new settings by executing *make* like earlier.

3 How to build Inviwo with ENVISIoN on Windows 10

These instructions show how to install and build Inviwo and ENVISIoN on Windows 10.

3.1 Dependencies

A dependency is a program or software that needs to work because the program you are installing relies on it. To be able to install Inviwo the following dependencies needs to be installed. Inviwo will only compile with Windows 64bit, so make sure all dependencies are also installed as 64bit.

- **Anaconda**
  Download [Anaconda] `https://www.anaconda.com/distribution/#windows` with python 3.7.
- **Qt**
  Download [Qt] (`https://www.qt.io/download`) opensource. Version 5.14.2 is tested and recommended. Make sure you select install option MSVC-2017 64-bit.
- **CMake**
  Download and install [Cmake] (`https://cmake.org/download/`).
- **Visual Studio 2017**
  Download [Visual Studio 2017] (`https://visualstudio.microsoft.com/vs/older-downloads/`). So far, Inviwo has not been able o compile with Visual Studio 2019.

3.2 Setup project

Open the Anaconda Prompt from your Windows searchbar. Then install the following dependencies by writing this in your prompt:

```
conda install git pybind11 numpy scipy matplotlib markdown regex
   wxpython h5py hdf5 libpng libtiff jpeg cmake nodejs
```

Then navigate to your home directory and run the following commands:

```
mkdir ENVISIoN
cd ENVISIoN
```

Then you are going to clone Envision when standing in the ENVISIoN directory by running this command:

```
git clone https://github.com/rartino/envision
```

Next step is to clone Inviwo. Run the following command while still in the ENVISIoN directory:

```
git clone https://github.com/inviwo/inviwo
```

Then go to the inviwo directory, switch to the branch v0.9.10 and update the registered submodules by running the following commands:

```
cd inviwo
git checkout v0.9.11
git submodule update --init --recursive
```

After this we want to patch Inviwo. Make sure you are in the inviwo directory and run:

```
git apply ../ENVISIoN/inviwo/patches/2019/transferfunctionFix.patch
git apply ../ENVISIoN/inviwo/patches/2019/deb-package.patch
git apply ../ENVISIoN/inviwo/patches/2019/paneProperty2019.patch
git apply ../ENVISIoN/inviwo/patches/2019/sysmacro.patch
git apply ../ENVISIoN/inviwo/patches/2019/inviwo-v0.9.10-extlibs.patch
```

3.3 CMake compiling

Create a build directory in the ENVISIoN folder that you made earlier. To do this make sure you are standing in the ENVISIoN directory and write:

```
mkdir inviwo-build
```

Then, open CMake (cmake-gui) and configure the projet with Inviwo as your source code and the new build directory inviwo-build as where to build the binaries. When configuring the project, make sure you select Visual Studio 2017 with x64 for platform option.

The following flags should be added to the default:

```
IVW_USE_EXTERNAL_HDF5:BOOL=OFF
IVW_IMG_USE_EXTERNAL:BOOL=ON
IVW_EXTERNAL_MODULES:PATH="{path to project}/ENVISIoN/inviwo/modules"
IVW_MODULE_CRYSTALVISUALIZATION:BOOL=ON
IVW_MODULE_FERMI:BOOL=ON
IVW_MODULE_GRAPH2D:BOOL=ON
IVW_MODULE_PYTHON3:BOOL=ON
IVW_MODULE_PYTHON3QT:BOOL=ON
IVW_MODULE_QTWIDGETS:BOOL=ON
IVW_MODULE_HDF5:BOOL=ON
IVW_PACKAGE_PROJECT:BOOL=ON
IVW_PACKAGE_INSTALLER:BOOL=ON
```

When done configuring press "Generate" and then open the project with Visual Studio 2017. In Visual Studio, set the build type in the upper menu bar to RelWithDebInfo and make sure that the solution platform is x64. Then press f5 (or fn + f5) and the building of the project should start. This can take some time. Check the Error List during the building to make sure nothing goes wrong. When the building is complete Inviwo should start.

3.4 Start ENVISIoN

Go back to your Anaconda Prompt and move to your projects directory (should be ENVISIoN) and run:

```
cd ENVISIoN
npm install
npm run start
```

Now Envision should start and the Envision GUI should pop up. For more information abut how to use ENVISIoN's GUI, see section 7. If you run into errors during the installation, see section 8 where some common errors are listed.

4 Preparing VASP-files before starting ENVISIoN

Before visualizations can be made of input data from VASP, the user have to make sure that VASP-files are made in correct format. This applies to a couple main files in VASP that the user has specified before their VASP-run and also after their VASP-run. The restrictions on the formating of VASP-files are a byproduct of the fact that ENVISIoN has a couple limitations in it's parsing of VASP-files. These limitations are by no way intentional and they are born from theoretical sources and from the time-restrictive nature of the project development itself.

4.1 Preparing for bandstructure visualization

Before deciding to do any visualization of bandstructures at all, be sure that the VASP-files OUTCAR and KPOINTS are available in the VASP-directory you wish to visualize from. Bandstructure visualization can only be made for eight types of brillouin zones if the user wishes to do so. Namely the following: Primitive Cubic (CUB), Body-Centered Cubic (BCC), Face-Centered Cubic (FCC), Primitive Tetragonal (TET), Primitive Hexagonal (HEX), Primitive Orthorhombic (ORC), Primitive Triclinic type 1a/2a (TRI1a/TRI2a) and Primitive Triclinic type 1b/2b (TRI1b/TRI2b). These brillouin zones have fixed symmetry k-points and are specified in an article by Wahyu Setyawan and Stefano Curtarolo [SC10].

If the user wish to visualize the bandstructure of any other type of brillouin zone, the parser for bandstructures will be unable to interpret the data from VASP.

Another requirement for a successful visualization of the bandstructure of a brillouin zone type is to format the KPOINTS-file in a specific way such as is illustrated in figure 1 below. Observe how every two coordinates represents a line between two high-symmetry points as chosen by the "line" in the third row, and how each of these pair of coordinates are separated by a blank row for aesthetical reasons. The parser can only handle reciprocal high-symmetry points so make sure that the fourth line in the KPOINTS-file has the word "reciprocal" written with no indentation, as shown in the figure.

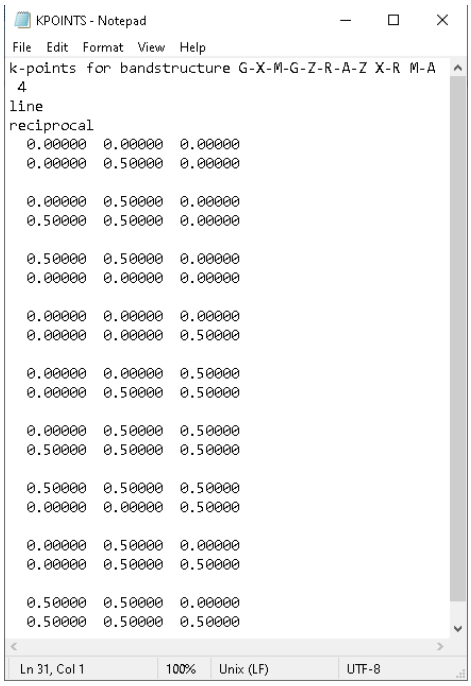

Figure 1: KPOINTS-file for bandstructure calculation of a TET brillouin zone.

4.2 Preparing for fermi-surface visualization

Make sure that a KPOINTS-file and a EIGENVAL-file is present in the VASP-directory you want to input in ENVISIoN.

When choosing to visualize fermi-surfaces, make sure that you use the format shown in figure 2 for the KPOINTS-file. During visualization of the fermi-surface of a crystal, we want a very big mesh of k-points and an optimal mesh is 10x10x10. Hence make sure that you the fourth row in KPOINTS is at least 10 on each position, as shown in the figure. A very important thing to also note is that the user has to supply the second row with only the a zero, as shown in the figure, as it will force VASP to generate a mesh in the size specified on the fourth row of the file during a VASP-run.

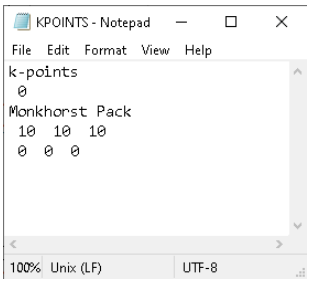

Figure 2: KPOINTS-file for fermi-surface visualization.

Another, very important, thing to note is that the fermi-surface visualization requires the flag ISYM in the INCAR-file of VASP to be set to the value -1 before a VASP-run. If ISYM is not set to -1, the parser for Fermi-surface will not be able to extrapolate the k-points that are given by VASP, since they are symmetric. If you strive to improve your visualization of the fermi-surface of a crystal, simply increase each integer on the fourth row on the figure above

from 10 and above. Observe that the fifth row must have zeroes in order to generate a mesh centered in a origin of the Monkhorst-Pack, without any shifts from this origin. This is for the sake of avoiding errors in the VASP-run.

4.3 Preparing for unitcell visualization

Make sure that the VASP-files POTCAR and POSCAR are available in your VASP-directory before you decide to visualize the unitcell.

For the unitcell visualization, it is very important how you have written the POTCAR-file in relation to the POSCAR-file. Say you have a crystal structure made of three atomic species and you want to do a VASP-run of it. You will first have to find the POTCAR-files for each atomic species, which is supplied by VASP itself. Then you concat them in order to create a single POTCAR-file which represent all atomic species in a single file. Now the dangerous and very common mistake to do in this case is that you concat the separate atomic species POTCARs in an order that does not consist with the order written in the POSCAR-file. This can lead to wrong conclusions of your VASP-calculations. Figure 3 examplifies the correct way of concatting POTCARs.

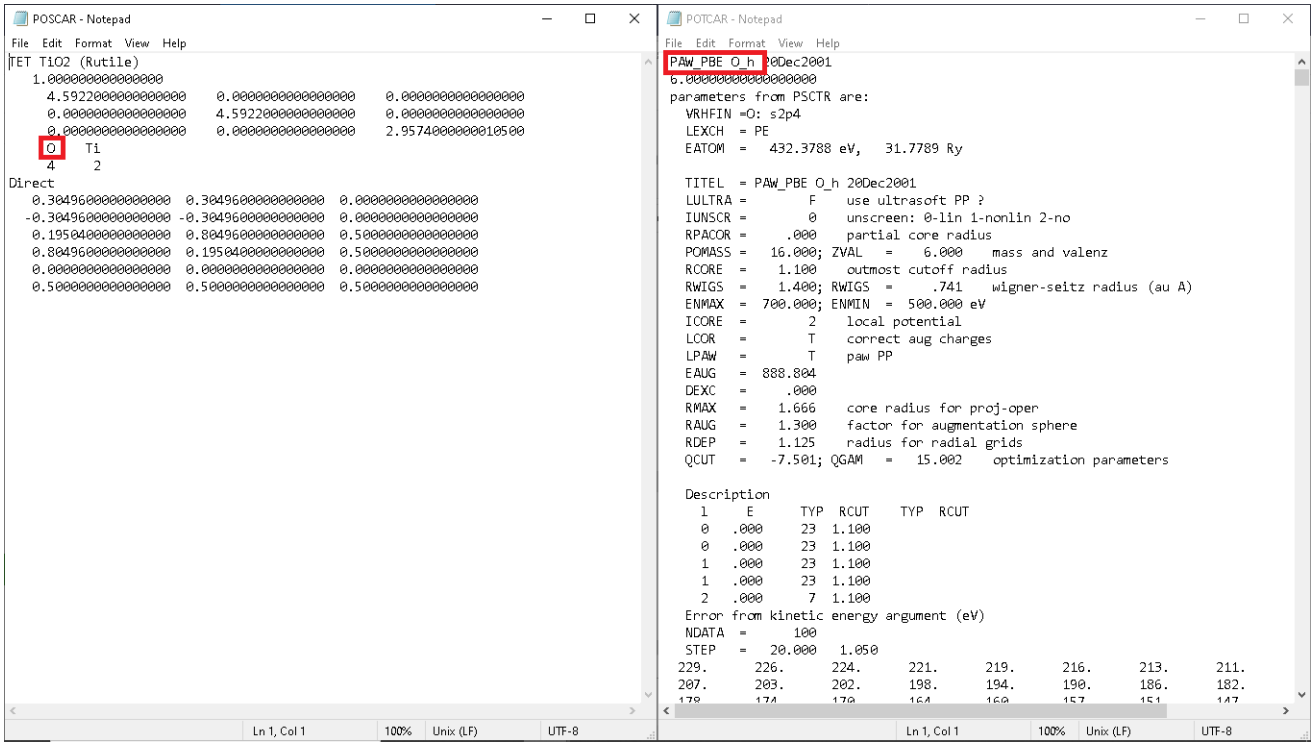

Figure 3: In this example with the crystal TiO2, the POSCAR-file has Oxygen written as the first atomic species (red box) and hence the common POTCAR has the pseudopotentials of Oxygen atoms concat first and the pseudopotentials of titanium atoms second.

To successfully visualize the unitcell, the user has to make sure that there is no indentation in the row where the user specifies if the atomic positions are given in direct or cartesian coordinates in the POSCAR-file. There is also a requirement that all values in the POSCAR-file, besides the row which supplies the number of atoms per atomic species (which are given in integer form), are given in decimal form (float form). Figure 4 below shows the correct format of POSCAR in

the example with Rutile. Notice how the order of the atomic positions are consistent with the order of the atomic species, which is emphasized by the red box for the O-atom positions and the blue box for the Ti-atom positions.

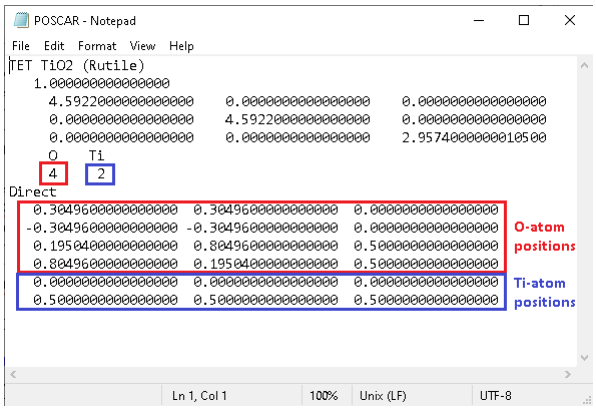

Figure 4: Example of the correct format of POSCAR-file with the crystal TiO2.

4.4 Preparing for DOS visualization

Make sure that the VASP-files POTCAR, POSCAR, INCAR and DOSCAR are available in your VASP-directory before you decide to visualize the density of states.

To ensure an error-free visualization of DOS, follow the directions mentioned in the section *Preparing for unitcell visualization*, since unitcell data is acquired simultaneously with DOS data with the VASP parser for DOS visualization.

4.5 Preparing for electron density visualization

Make sure that the VASP-files POTCAR, POSCAR and CHG are available in your VASP-directory before you decide to visualize the electron density. In the VASP-file INCAR, make sure that LCHARG flag is not set to False, in order to force VASP to write into the CHG-file during the VASP-run. Also make sure that ISYM flag in the VASP-file INCAR is not set to -1 before a VASP-run, which is done for the fermi-surface visualization, as this can cause a wrong visualization of the electron density.

To ensure an error-free visualization of electron density, follow the directions mentioned in the section *Preparing for unitcell visualization*, since unitcell data is acquired simultaneously with electron density data with the VASP parser for electron density visualization.

5 Start ENVISIoN or Inviwo

5.1 Start ENVISIoN

When the building for Inviwo with ENVISIoN is finished, we can start the application's graphical user interface. If you have structured the file system as explained in section ... and section

..., the start up of the application should be the same on every operating system. Open up a terminal or command window and navigate to the ENVISIoN folder located in the ENVISIoN directory. When you stand in .../ENVISIoN/ENVISIoN, execute the following command in the terminal:

```
npm run start
```

The application ENVISIoN should start and the GUI will pop up. For further instruction concering ENVISIoN's GUI, see section 7.

5.2 Start Inviwo

During the development of ENVISIoN it can be an advantage to use only Inviwo and its graphical user interface. If the installation of ENVISIoN with Inviwo is made correctly, open a terminal or command window and navigate to the ENVISIoN directory. Then, type in the following command in the terminal:

```
./inviwo-build/bin/inviwo
```

6 Start Inviwo and run ENVISIoN scripts

If the user wishes to run Inviwo with its own graphical user interface, it's possible and still have access to the visualizations provided by ENVISIoN. These visualizations are stored in the form of Python scripts that can be compiled through the Inviwo user interface.

To run Inviwo in an UNIX environment, execute the commands below.

```
cd ENVISIoN/ENVISIoN
 ./inviwo-build/bin/inviwo
```

When the Inviwo interface has opened, follow the instructions given in figure 5 and in the list below to run a visualization script.

1. Locate and press the Python menu in the Inviwo bar.
2. Open the Python editor by pressing it.
3. In the Python editor, click Open Script.
4. Select one of the scripts. The ENVISIoN scripts can be located in *~/ENVISIoN/scripts*.
5. Click open.
6. Click the button in the top left corner to run.

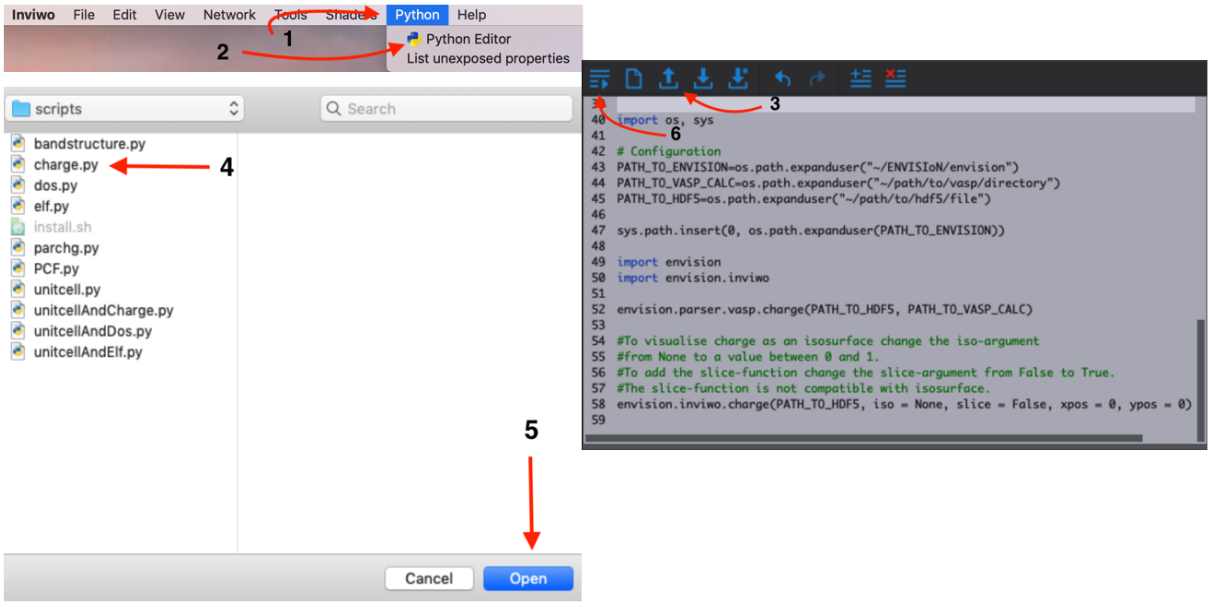

Figure 5: Cutout from Inviwo with instructions on how to run a ENVISIoN visualization script in numeric order.

7 Using ENVISIoN's GUI

ENVISIoN is equipped with a graphical user interface to simplify the usage of ENVISIoN.

7.1 Start-up

When the user starts the application through a computer terminal (see chapter 5) a window opens, see figure 7 for Ubuntu and figure for Windows. The start-up window of ENVISIoN has a sidebar menu to the left with different choices. Each choice has its own content to the right. By default, the "Dataset loader", is chosen from start. The other possible choices are "Parser" or "About". Under "Active datasets" the loaded datasets will appear. Each alternative will be explained in their own sections in this chapter. The following figures of the GUI in this chapter will be for Windows but the content is the same for every operative system.

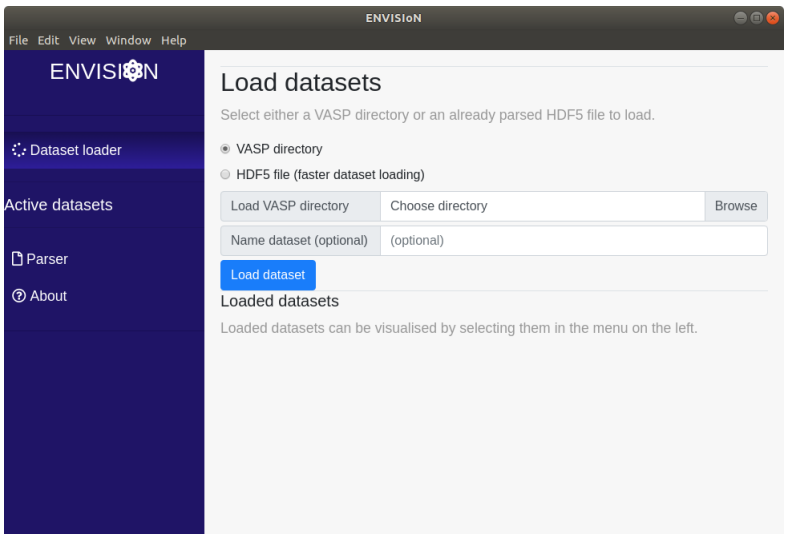

Figure 6: ENVISIoN start-up window for Linux.

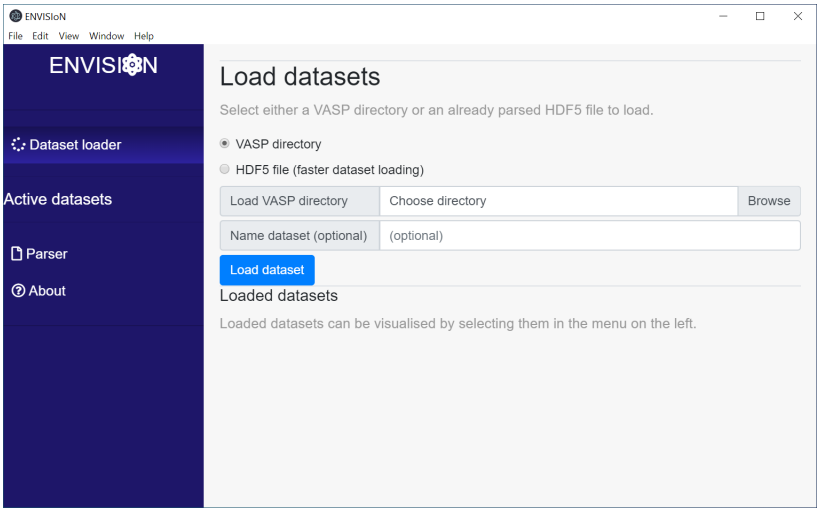

Figure 7: ENVISIoN start-up window for Windows.

7.2 Dataset loader

In the "Dataset loader" a user can select either a VASP file or a HDF5 file to load which then will be visualised. To load a VASP file, the box "VASP directory" need to be checked. To load a HDF5 file the box "HDF5 file" need to be checked.

For a quick step-by-step guide, scroll down to section 7.2.3.

7.2.1 Load a VASP file

For loading a VASP file, the content for the "Dataset loader" is shown below in figure 8.

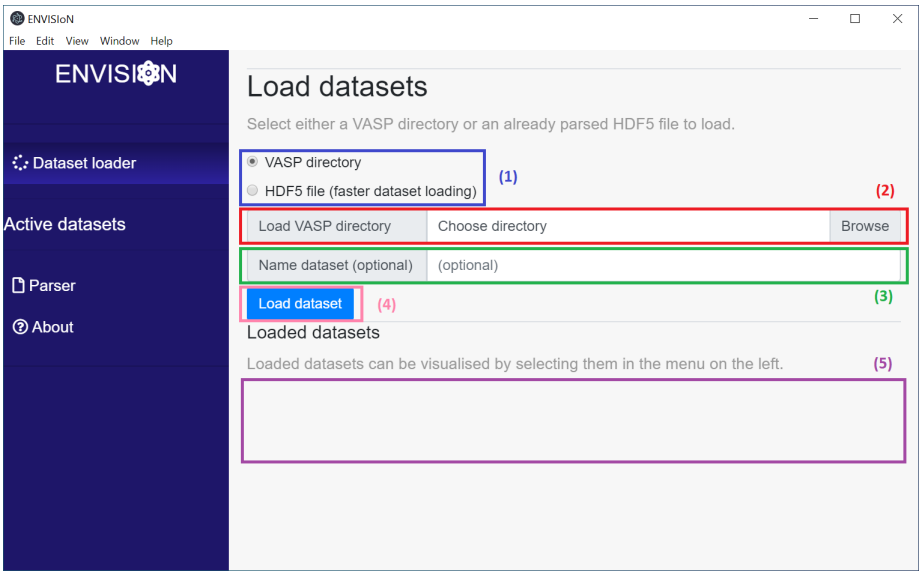

Figure 8: Dataset loader for VASP source.

In the blue box labeled (1) the box "VASP directory" need to be checked. In the red box labeled (2) the path to the VASP directory to load is selected. By clicking on "Browse" a new window will appear where the user navigates to the VASP directory on the computer and selects it. In the green box labeled (3) the user can write a name for the dataset. This is optional. In the pink box labeled (4) is the "Load dataset" button. When pressing this button the dataset will be loaded. The loaded dataset will be displayed in the purple box labeled (5) and in the sidebar menu under "Active datasets".

7.2.2 Load a HDF5 file

For loading a HDF5 file, the content for the "Dataset loader" is shown below in figure 9.

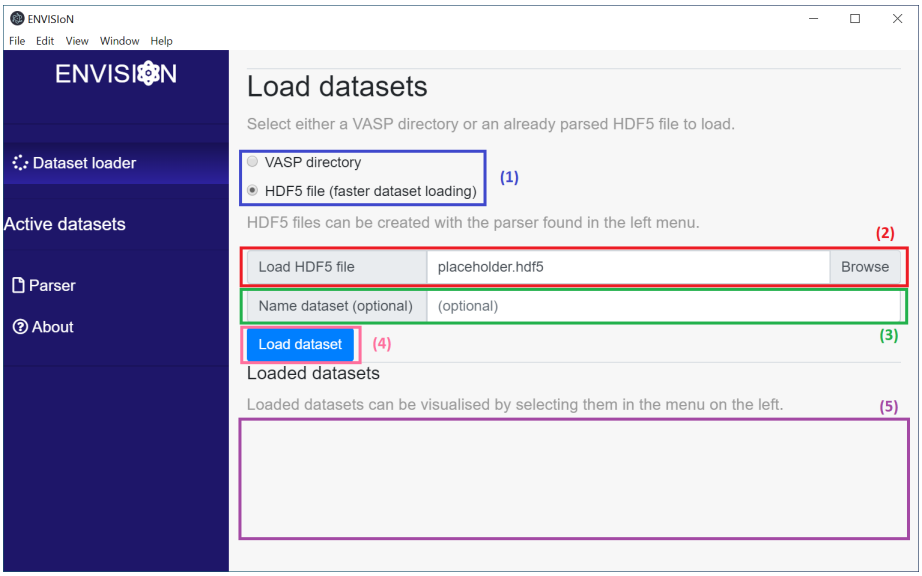

Figure 9: Dataset loader for HDF5 source.

In the blue box labeled (1) the box "HDF5 file" need to be checked. In the red box labeled (2) the path to the HDF5 file to load is selected. By clicking on "Browse" a new window will appear where the user navigates to the HDF5 file on the computer and selects it. In the green box labeled (3) the user can write a name for the dataset. This is optional. In the pink box labeled (4) is the "Load dataset" button. When pressing this button the dataset will be loaded. The loaded dataset will be displayed in the purple box labeled (5) and in the sidebar menu under "Active datasets".

7.2.3 Quick Step-by-step guide

To load a VASP-file (referring to figure 8):

1. Check the box "VASP directory" in (1).
2. Choose a VASP directory on your computer by clicking "Browse" in (2).
3. (optional) Name the dataset by writing in (3).
4. Click "Load dataset" in (4).
5. Now the loaded dataset is showing in (5) and in the sidebar menu under "Active datasets".

To load a HDF5 file (referring to figure 9):

1. Check the box "HDF5 file" in (1).
2. Choose a HDF5 file on your computer by clicking "Browse" in (2).
3. (optional) Name the dataset by writing in (3).
4. Click "Load dataset" in (4).
5. Now the loaded dataset is showing in (5) and in the sidebar menu under "Active datasets".

7.3 Active datasets

When a dataset is loaded for visualisation it will appear under “Active datasets” in the sidebar menu.

7.3.1 Starting the visualisation

By clicking on one of datasets under “Active datasets”, new content will show up to the right, see figure 10 below. In figure 10 the dataset “BaSO4_ORC” is loaded as an example.

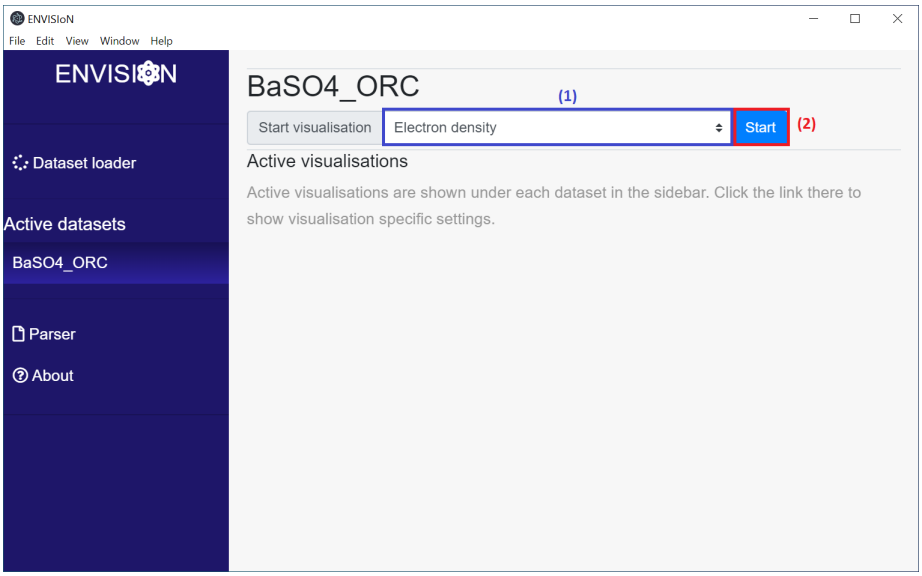

Figure 10: Start the visualisation. Here the dataset BaSO4_ORC is loaded as an example.

By pressing the area inside the blue box labeled (1) the user can select which visualisation type to visualise from the drop-down menu. The different possible visualisation types are:

- Electron density
- Unitcell
- Bandstructure 3D
- Fermi surface

In the red box labeled (2) is the “Start” button. When pressing this button the visualisation will start for the chosen visualisation type. The visualisation will appear under the loaded dataset in the sidebar menu. When pressing the visualisation in the sidebar menu new content with visualisation controls will appear to the right. Also the canvas/canvases belonging to the selected visualisation type will pop up next to the GUI.

7.3.2 Visualisation controls

When clicking on the ongoing visualisation of a dataset in the sidebar, visualisation controls will appear to the right. These controls are different for different visualisation types. There are only controls for the visualisation types “Electron density” and “Fermi surface”.

Electron density controls
If the ongoing visualisation is “Electron Density” the controls for interacting with the visualisation are shown in figure 11 below.

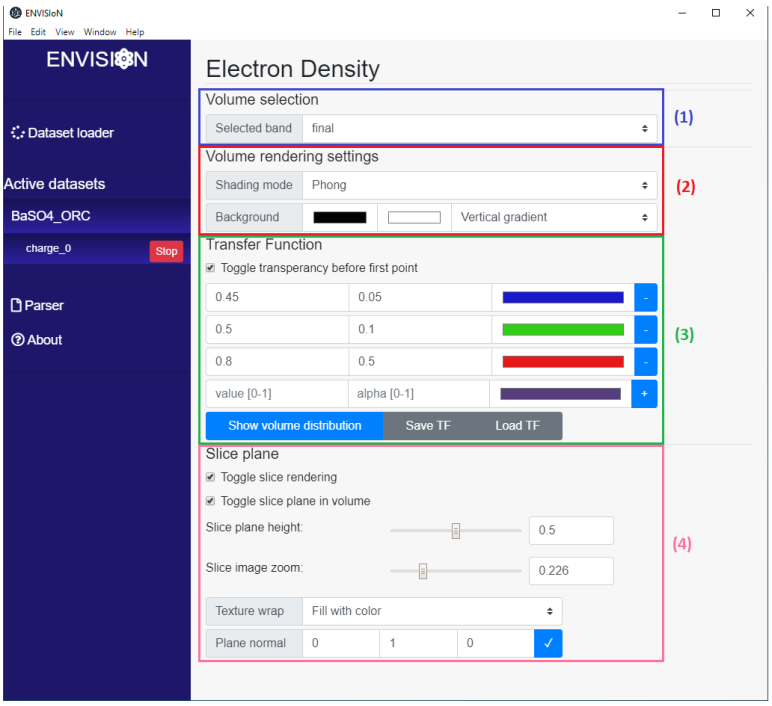

Figure 11: The content for the electron density visualisation controls.

For electron density the user can control the volume selection by selecting band, the volume rendering settings, the transfer function by visualising certain density intervals with different colors and the slice plane to see the electron density in a 2-dimensional plane. In the blue box labeled (1) the user can regulate the volume selection by selecting which band to visualise from a drop-down menu. In the red box labeled (2) the user can select the Shading mode from a drop-down menu and also change the Background color and gradient. In the green box labeled (3) the user can change the transfer function. This is a way to choose different colors for different electron density intervals. The intervals are normalized. The user can also select the blue “Show volume distribution” button and then a new diagram will pop up containing information about the volume distribution for the selected dataset. In the pink box labeled (4) the user can configure the slice plan to see a 2-dimensional cross-sectional area of the electron density connected to the 3-dimensional electron density.

Fermi surface controls
If the ongoing visualisation is “Fermi surface” the controls for interacting with the visualisation are shown in figure 12 below.

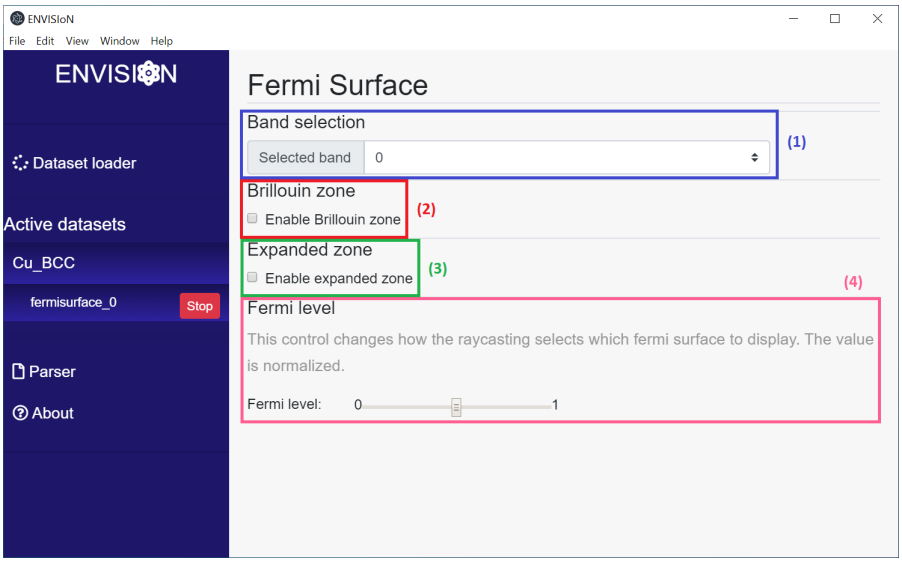

Figure 12: The content for the fermi visualisation controls.

For fermi surface the user can control which band that should be visualised, enable the Brillouin zone, enable expanded zone and regulate the fermi level. In the blue box labeled (1) the user can select which band to visualise from a drop-down menu. When starting the visualisation the current band will be selected automatically and also the correct fermi level (in the pink box labeled (4)) for each band is set to default. In the red box labeled (2) the user can enable or disable the Brillouin zone by having the checkbox checked or unchecked. In the green box labeled (3) the user can enable or disable expanded zone by having the checkbox checked or unchecked. In the pink box labeled (4) the user can control the fermi level by dragging the slider to the right or to the left. The scale is normalized between 0 and 1.

7.4 Parser

If you choose “Parser” in the sidebar menu, new content will be exposed to the right, see figure 13 below.

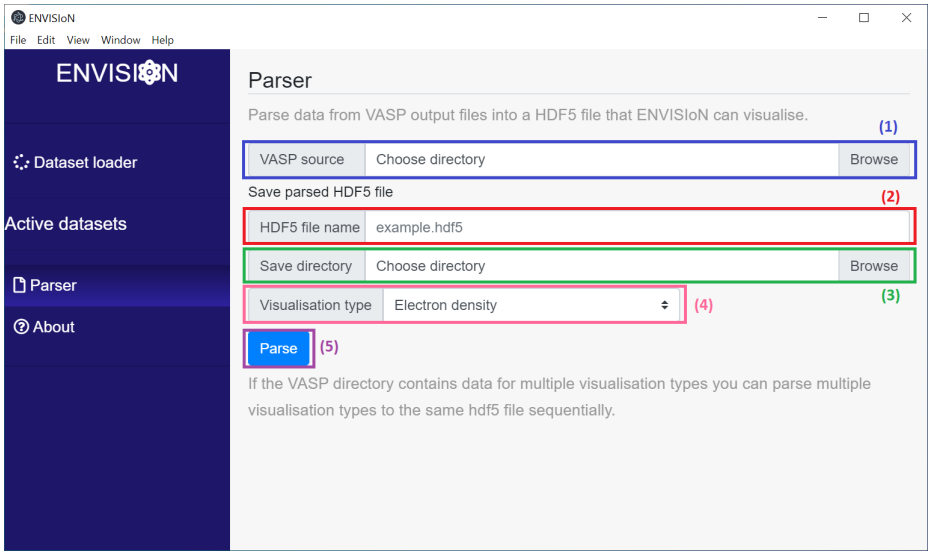

Figure 13: The content for the parser menu.

For a quick step-by-step guide, scroll down to section 7.4.1

In figure 13, in the blue box labeled (1), the path to the directory of VASP files to parse is selected. By clicking on "Browse" a new window will appear where the user navigates to the directory of the VASP files on the computer and selects it. In the red box labeled (2) the user writes the name for the new HDF5 file, it needs to end with .hdf5. In the green box labeled (3) the path to the saving directory of the new HDF5 file is selected. When clicking on "Browse" a new window will appear where the user navigates to the saving directory and selects it. In the pink box labeled (4) the user can select which visualisation type to parse from a drop-down menu. The possible types to parse are:

- Electron density
- Bandstructure
- Unitcell
- Fermi surface

In the purple box labeled (5) is the "Parse" button. By pressing this button the parser will parse the VASP files and create a HDF5 file in the saving directory.

7.4.1 Quick Step-by-step guide

Parsing VASP files to a HDF5 file:

1. Enter path to VASP directory in (1).
2. Write name for the HDF5 file in (2).
3. Enter path to saving directory in (3)
4. Select type in (4).
5. Press "Parse" in (5).

7.5 About

When selecting "About" in the sidebar menu there will be new content to the right that contains brief information about ENVISIoN and the project members through the years.

7.6 If ENVISIoN does not respond or crashes

If the user have multiple ongoing visualisation or if the user has stopped one visualisation and then want to start another the GUI can stop responding or sometimes even crash. If the GUI falls behind in the number of requests from the user, it will stop responding and a pop up in the GUI will show. The pop up contains the message "Loading... envision is requests behind". See figure 14 below.

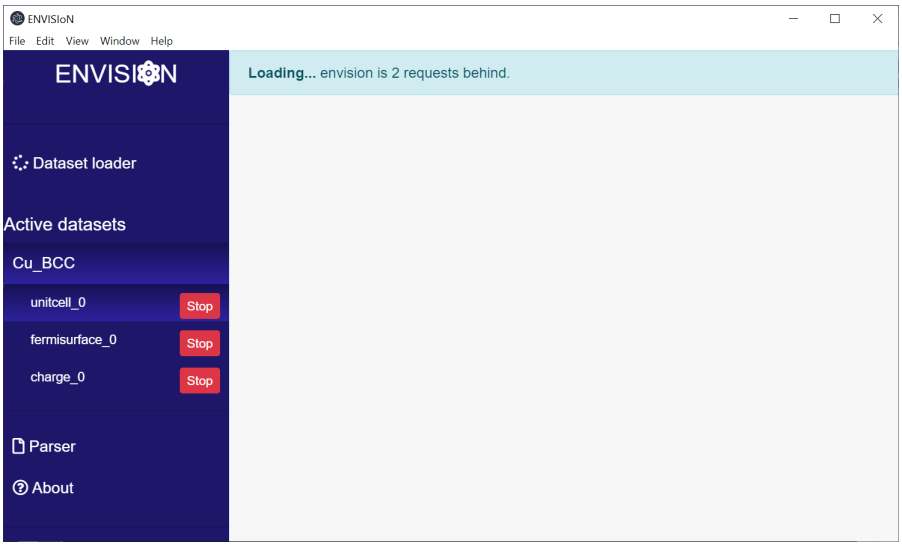

Figure 14: The pop up when ENVISIoN is a number of requests behind.

To recover from that ENVISIoN is a number of requests behind you can click on “View” in the upper menu and then “Reload” to reload the GUI. Then you need to start from the beginning and load the dataset once again.

If ENVISIoN is a number of request behind and the user continues to click around in the GUI and tries to load a new dataset or create a new visualisation the GUI will crash. If it does the following message box will pop up, see figure 15.

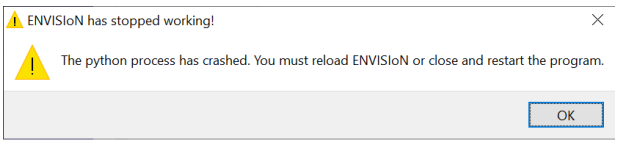

Figure 15: The message box when ENVISIoN crashes.

To recover from that ENVISIoN crashes you can either reload the page as mentioned above, or just close the GUI window and restart ENVISIoN.

8 Common errors

8.1 Install Qt

Inviwo uses the graphics library Qt which isn't always installed properly. These instructions show how to download and install the latest version of Qt on Ubuntu 10.04 LTS. That is, in the moment of writing this user guide, version 5.12.3.

To download the installation file into the */Downloads* directory, simply execute the commands below.

```
cd ~/Downloads
wget http://download.qt.io/official_releases/qt/5.12/5.12.3/qt-
   opensource-linux-x64-5.12.3.run
```

When the installation file has finished downloading, the user won't have permission to run the file. To change permissions and run the file by executing the commands below and enter your superuser password immediately after.

```
chmod +x qt-opensource-linux-x64-5.12.3.run
sudo ./qt-opensource-linux-x64-5.12.3.run
```

An Qt installer is now shown on the screen. Notice that the manual installation will force a installation of the Qt editor as shown in step 6. The entire installation will occupy approximately 5.12 GB. Follow the instructions in figure 16 to complete the installation.

After the installation is done, the path to Qt needs to be added to the system. Add the necessary paths by executing the commands below.

```
cd /usr/lib/x86_64-linux-gnu/qtchooser
sudo echo "/opt/Qt5.12.3/5.12.3/gcc_64/bin" | sudo tee -a default.conf
sudo echo "/opt/Qt5.12.3/5.12.3/gcc_64/lib" | sudo tee -a default.conf
```

The system is now ready for an Inviwo installation.

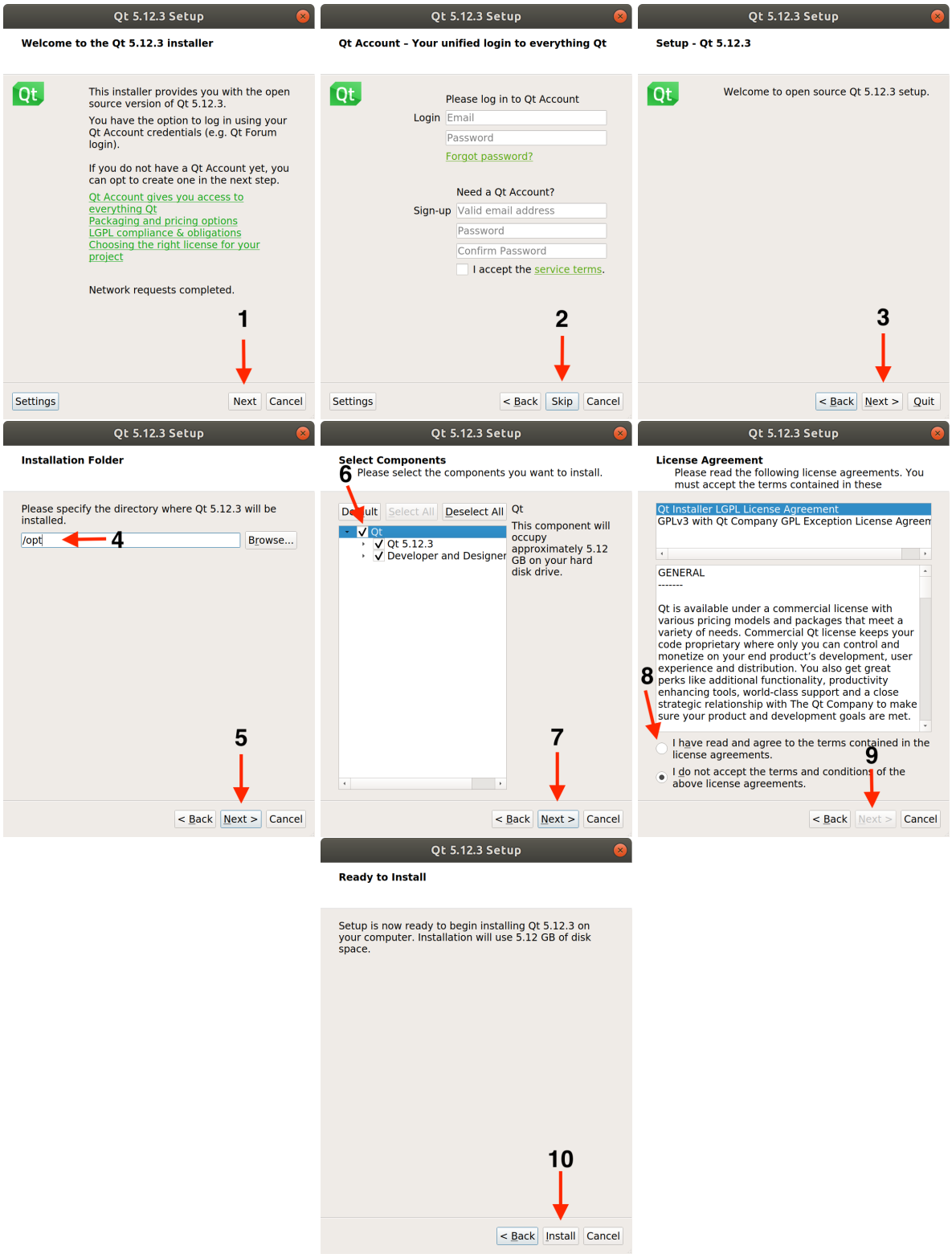

Figure 16: Instructions for installation of Qt 5.12.3.

8.2 Set the INVIWO_HOME path

Before using ENVISIoN's GUI it is good to check that your environmental variable INVIWO_HOME is set to the correct value. Otherwise the system will not find the modules inviwopy or inviwopy-

app. If you have started to use the GUI you will receive the following message if the variable is not set:
“Module error: No module named ’[The missing module]’. Can not find module. Please check that the environment variable INVIWO_HOME is set to the correct value in the computers system settings.”

8.3 The module Inviwopyapp did not build

If the inviwopyapp did not build when building in Visual Studio, open the project once again in Visual Studio 2017 and make sure “inviwopyapp” is listed to the right in the “Solution Explorer”, see figure 17. Inviwopyapp should be under “minimals” in the list.

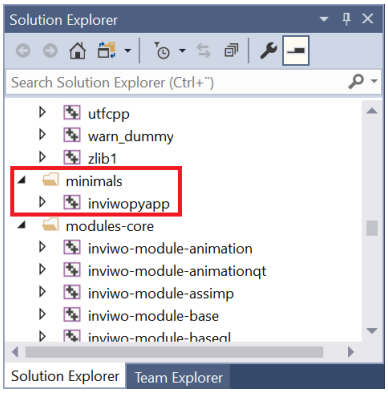

Figure 17: The “Solution Explorer” in Visual Stuido 2017

Now try to build the project again but by choosing “Build” in the upper menu and then “Build Solution” instead of pressing fn + f5. The build should start, keep track of possible errors during the building. When it’s finished, check that inviwopyapp had built.

References

[SC10] Wahyu Setyawan and Stefano Curtarolo. “High-throughput electronic band structure calculations: Challenges and tools”. In: *Computational Materials Science* 49.2 (2010), pp. 299–312. DOI: https://doi.org/10.1016/j.commatsci.2010.05.010.

A Licens