An Informal Guide to DASVader.jl and the Julia Language

Hello there!

Welcome to DASVader.jl!

As I worked with Julia, I started writing a bunch of code to handle the Distributed Acoustic Sensing (DAS) data from the FIMOPTIC project at IPGP. After a few months of coding and testing, I realized I had enough tools to perform the basic steps one would typically follow in SAC when working with seismic data: preprocessing, filtering, plotting, and saving. So, I decided to bundle everything into this package so that everyone can benefit from it!

About This Guide

This PDF is an informal and informative guide. It's not part of the formal documentation that will eventually accompany DASVader.jl. Instead, it's meant to help anyone get started with the package quickly.

I also place special emphasis on using Julia, as I know it's not the most well-known computational language out there (yet!).

What to Expect

We'll start by installing Julia on your computer and setting up DASVader.jl. After that, I'll give you a quick tour of the Julia environment (don't worry—I'll also share links to better tutorials than my own!). Finally, we'll go through the examples included with the package so you can follow along and start working with your own data.

Let's dive in!

1. Installing Julia Language.

Julia is a relatively new computational language designed specifically for science! It was created by scientists for scientists. When it first launched, it carried the slogan: "Julia: come for the syntax, stay for the speed." The idea behind Julia is to combine the ease of use and learnability of "modern" computational languages like Python and MATLAB with the speed, power, and flexibility of "classic" languages like C, C++, and Fortran. Julia excels at parallel computing, GPU programming, and machine learning, making it an incredibly versatile tool.

If this sparks your curiosity, now might be the perfect time to give it a try!

To get started, let's install Julia on your computer. Visit the official Julia downloads page and select the appropriate installer for your operating system.

https://julialang.org/downloads/#current_stable_release

I recommend you install version 11.1.

If you have **Linux** you should be able to use this command to get it immediately:

curl -fsSL https://install.julialang.org | sh

For **macOS**, you'll need to choose the installer that matches your machine's architecture:

- If your Mac uses an Intel processor, select macOS x86 (Intel or Rosetta).
- If your Mac has an M1 or M2 chip, choose macOS (Apple Silicon).

Once downloaded, open the .dmg file and drag the Julia app into your Applications folder. To use Julia from your terminal, you'll need to add it to your system's PATH. The exact steps depend on your macOS setup, but the path you're looking for is likely:

/Applications/Julia-1.11.app/Contents/Resources/julia/bin

Here's how to add it to your PATH:

Open your terminal and check which shell you're using by running:

echo $SHELL

If you're using zsh (the default in recent macOS versions), you'll need to edit the .zshrc file.
For older systems using bash, edit the .bash_profile file instead. Open the configuration file with a text editor, for example:

```
nano ~/.zshrc
```

Add the following line at the end of the file:

```
export PATH="/Applications/Julia-1.x.app/Contents/Resources/julia/bin:$PATH"
```

Save and close the file (in nano, press CTRL+O, then CTRL+X).

Refresh your shell:

```
source ~/.zshrc
```

Now, you should be able to launch Julia by simply typing julia in your terminal!

For **Windows**, there are installers available on the Julia Downloads page. Unfortunately, I haven't personally tried them, so I can't provide detailed steps. But don't worry! The installation process should be straightforward—just follow the instructions on the page.

A Note About JuliaUp

If you think Julia might be the right tool for you, consider checking out JuliaUp. It's the official way to install and manage Julia versions, making updates and version switching easier. You can learn more about it here.

https://github.com/JuliaLang/juliaup

However, fair warning: JuliaUp is a bit more hands-on and technical. That's why I've stuck with the "standard installation" instructions for this guide.

2. Getting started with Julia Language.

Once you have installed julia and added it to your path you should be able to launch julia from your terminal. Open a terminal and write:

julia

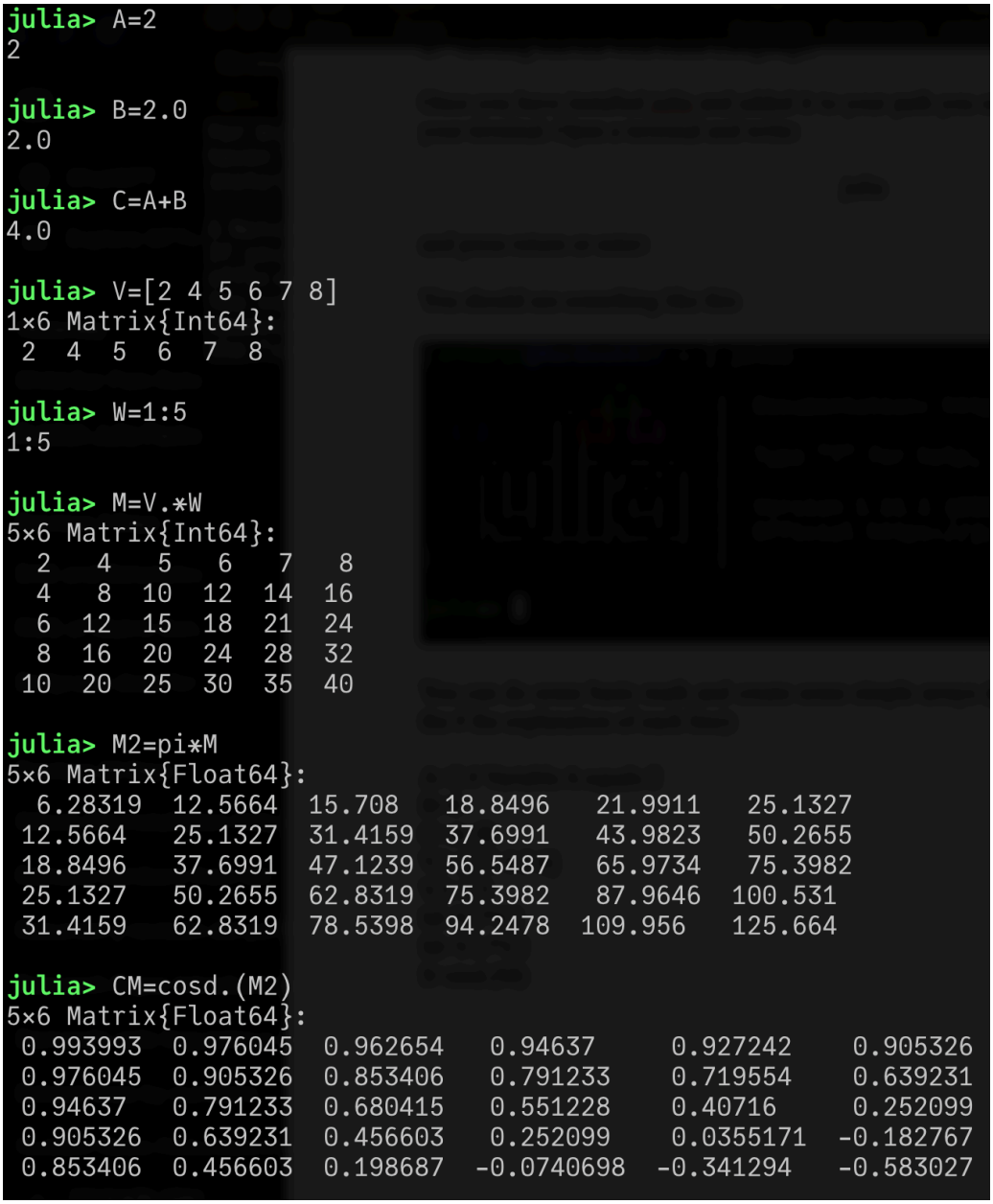
and press return or enter:

You should see something like this

You can do some basic math and create some simple arrays (hit return after each line and after the # the explanation of each line):

- A=2 # Variable A equals 2 (this is an Int64)
- B=2.0 # Variable B equals 4.0 (this is an Float64)
- C=A+B # Add A and B to get the answer (it is Float64)
- V1=[2 3 4 5 6] # Create a 1 row matrix (if you separate the number with ";" it is a vector)
- V2=1:5 # Create a range from 1 to 5
- M=V1.*V2 # vector multiplication. Notice we use a dot .* without the dot, give it a try without
- M2=pi*M # multiply M by pu
- CM=cosd.(M2) # get the cosine for each value in M2

Here a view of the same inside Julia



```
julia> A=2
2

julia> B=2.0
2.0

julia> C=A+B
4.0

julia> V=[2 4 5 6 7 8]
1×6 Matrix{Int64}:
 2  4  5  6  7  8

julia> W=1:5
1:5

julia> M=V.*W
5×6 Matrix{Int64}:
  2   4   5   6   7   8
  4   8  10  12  14  16
  6  12  15  18  21  24
  8  16  20  24  28  32
 10  20  25  30  35  40

julia> M2=pi*M
5×6 Matrix{Float64}:
  6.28319  12.5664  15.708   18.8496   21.9911   25.1327
 12.5664   25.1327  31.4159  37.6991   43.9823   50.2655
 18.8496   37.6991  47.1239  56.5487   65.9734   75.3982
 25.1327   50.2655  62.8319  75.3982   87.9646  100.531
 31.4159   62.8319  78.5398  94.2478  109.956   125.664

julia> CM=cosd.(M2)
5×6 Matrix{Float64}:
 0.993993  0.976045  0.962654   0.94637     0.927242    0.905326
 0.976045  0.905326  0.853406   0.791233    0.719554    0.639231
 0.94637   0.791233  0.680415   0.551228    0.40716     0.252099
 0.905326  0.639231  0.456603   0.252099    0.0355171  -0.182767
 0.853406  0.456603  0.198687  -0.0740698  -0.341294   -0.583027
```

I know this is very very basic, but you can see that working in julia is not very different from Python, R or Matlab (in which you might have some experience). If you want to dive more check out this links:

https://www.datacamp.com/tutorial/julia-programming-tutorial-for-beginners

For a more formal version that dives deeper into things try this PDF:

https://www.sas.upenn.edu/~jesusfv/Chapter_HPC_8_Julia.pdf

For a formal introduction to the language I have to recommend the tutorial from JuliaAcademy. You can install the full GitHub project in your computer:

https://github.com/JuliaAcademy/JuliaTutorials

Or you can go inside the introductory course and see the Jupyter notebooks (.ipynb one by one).

https://github.com/JuliaAcademy/JuliaTutorials/tree/main/introductory-tutorials/intro-to-julia

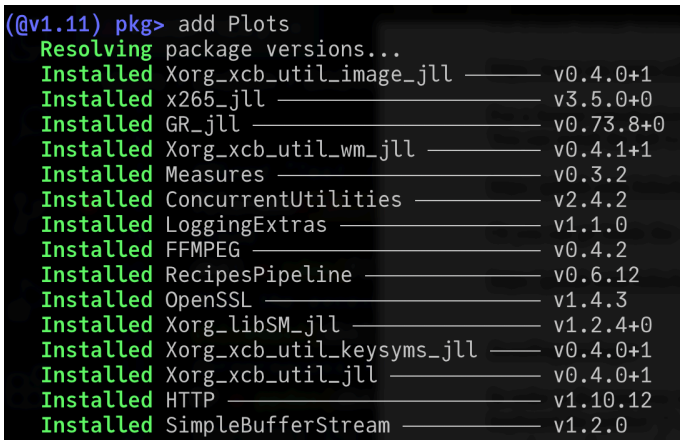
Installing packages

In order to do more you will need to install some packages. For this you will need to find in your keyboard the symbol "]". When you press this key into julia the repl (the Julia terminal changes) goes into package mode:

Here you can install any package. try installing a plotting library:

```
add Plots
```

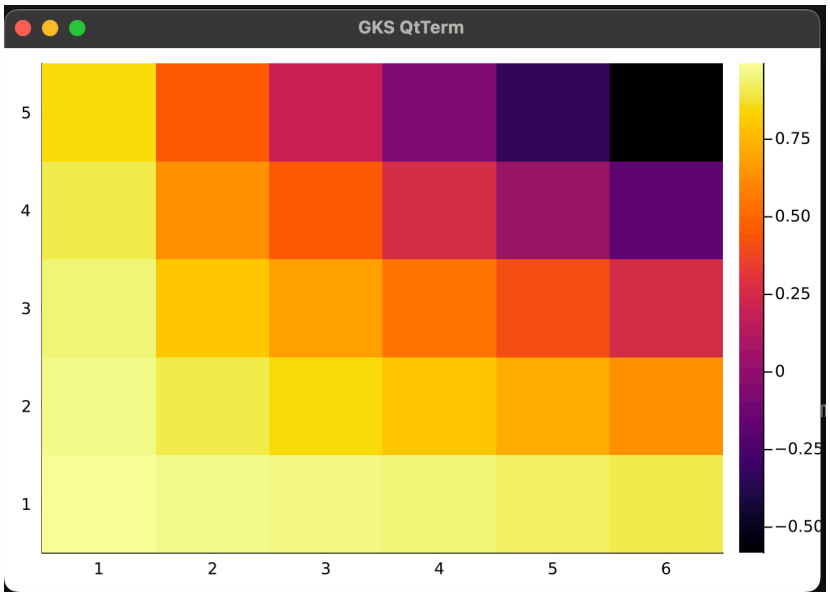
You should see something like this but longer:



Once its finished you should have Plots.jl in your system and you can do your first plot! just try

```
heatmap(CM)
```

A little window should open and show you this:



A word of warning Plots.jl is very powerful, maybe you can dive into this website if you are interested

https://docs.juliaplots.org/stable/

Using Shell mode!

Yes, Julia can give use access you a standard linux shell, you can change path, make dir, ls etc. For this just press ";" and the terminal will change:

```
shell> pwd
/Users/andurin

shell> mkdir Hello

shell> ls
09_05                                Untitled1.ipynb
DASVader_V0                          Untitled2.ipynb
DASVader_V0.1.zip                    Voids_Model_equi_SlowLen_2.txt
```

If ever in doubt press "CONTROL and C" to kill a line and "CONTROL and D" to kill Julia

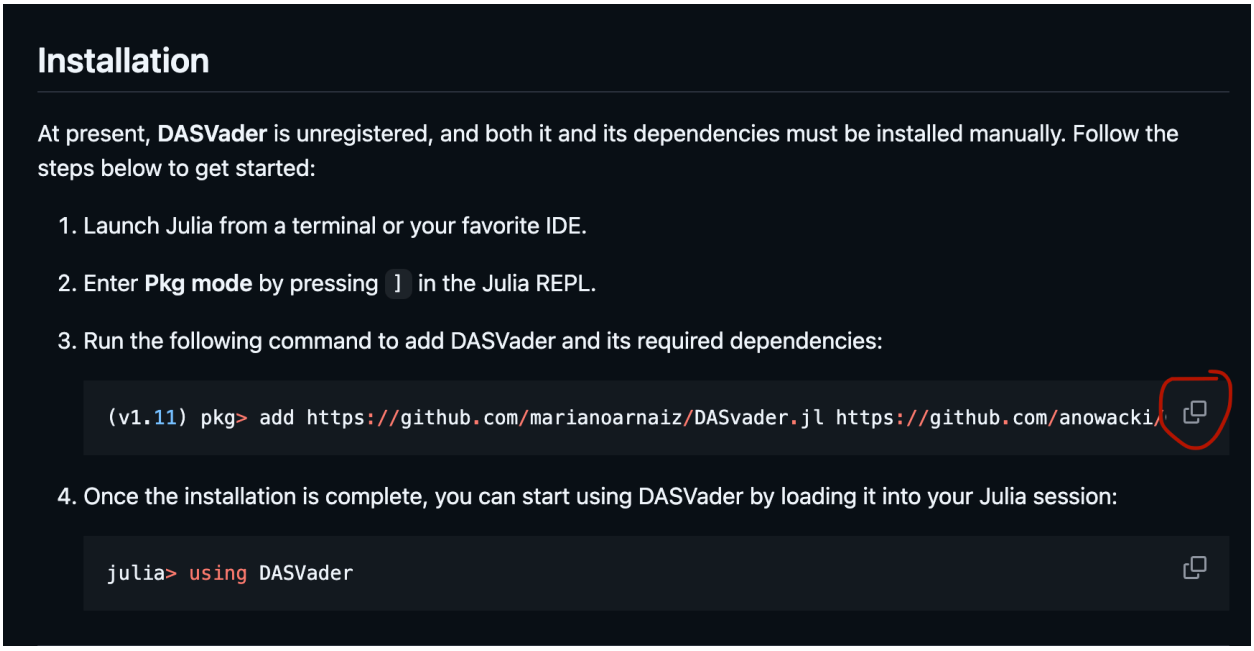
3. Install DASVader

Now you are ready to install DASVader.jl!

Now, before anything, you can visit the project's GITHUB page here:

https://github.com/marianoarnaiz/DASvader.jl

There you can copy the following line if it does not work from this PDF. Go to Installation and copy the like in point 3. Just click on the 2 little squares to copy the line and paste is in julia.



If not, open the package manager (using "]") and paste the next instruction.

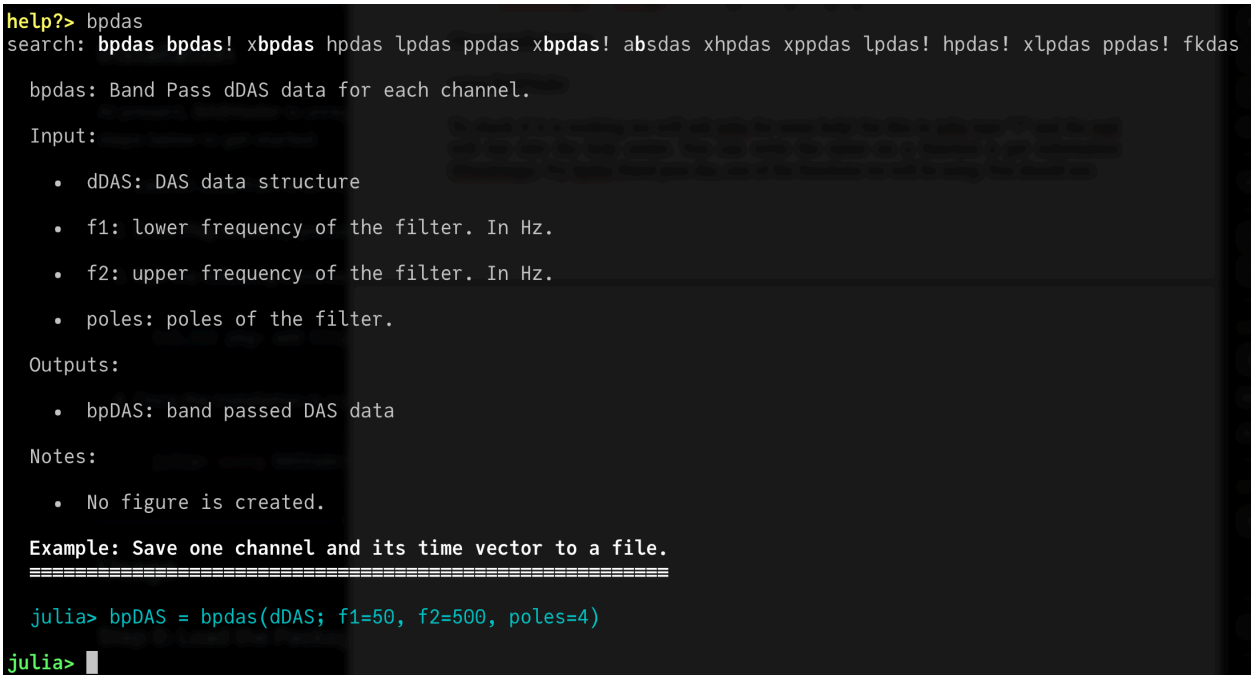
add https://github.com/marianoarnaiz/DASvader.jl https://github.com/anowacki/Geodesics.jl https://github.com/anowacki/Seis.jl

Notice that it is much more complicated that installing Plots.jl. The reason is that DASVader as well as Geodetics.jl and Seis.jl are not registered packages yet.

Once installed write:

using DASVader

To check if it is working we will ask julia for some help! for this in julia type "?" and the repl will tun into the help center. You can write the name on a function a get information (Docstrings). Try bpdas (band pass das, one of the functions we will be using. You should see:



OK. This should get you started. Now you can go to the examples I have provided.

BEFORE YOU START TO GO THROUGH THE EXAMPLES PLEASE GET SOME DATA! I HAVE UPLOADED SOME FILES TO DROPBOX. Here are the links

A Noisy blast file:

https://www.dropbox.com/scl/fi/c6ui9cxcb1gxawm0qqkcp/SR_DS_2023-08-24_14-06-17_UTC_Noisy_Blast.h5?rlkey=e3yn1likn3mkrhesyx85pcpw8&st=6zh6w1i0&dl=0

A file with a micro event:

https://www.dropbox.com/scl/fi/xxrd8rlw8kwthmfgwamwx/SR_DS_2023-10-30_12-01-40_UTC_Microevent.h5?rlkey=3zjvn706s46grco4gzhzrqmu5&st=0di6y3x8&dl=0

A file with a big blast:

https://www.dropbox.com/scl/fi/abcb1zphevctfkzetn7ql/SR_DS_2024-10-22_14-08-02_UTC_Big_Blast.h5?rlkey=mi9mvj3ynzptgxj3e1khkre8l&st=46eo4s9h&dl=0

A file with something that might be an event:

https://www.dropbox.com/scl/fi/n5czzuez7lq2yt3j2s5k5/SR_DS_2024-10-22_21-27-02_UTC_Hidden_Event.h5?rlkey=fsb9tq7wuxgbek5av3uc8rpzd&st=ncrqbwhx&dl=0

I suggest you get a few of these, put them in a directory and open each example in a text editor. You can begin just by copying and pasting lines from each example file into Julia. If you use VSCODE, VIM or EMACS you should really check this links out:

https://www.julia-vscode.org/

https://github.com/JuliaEditorSupport/julia-vim

https://github.com/JuliaEditorSupport/julia-emacs