YTS_tracking

This is what I found
NOTE THAT: I've added a label on top of each plot to say if its GADI output or LOCAL output

1. `runme.py`

NO `yts` factor anywhere so there is no conversion

2. `IsmipF.py`

there is a `yts` conversion when I define friction parameters, In Pattyn 2008: The A value is given in $Pa^{-1}\ a^{-1}$ so as per the issm website version of this code I convert to $Pa^{-1}\ s^{-1}$.
I do it like this:

```
# convert to ISSM units:
A_seconds =  A / md.constants.yts
c = 1.0
beta_squared = 1.0 / (c * A_seconds * H_0)
    md.friction.coefficient = np.sqrt(beta_squared) *
np.ones((md.mesh.numberofvertices))
```

Note: have no functions in neither of these scripts just the if statements so my code follows the code in the issm website examples.

Once I get my `.bin` and `.toolkits` I run those on gadi using a queue file that looks like this:

3. `IsmipF.queue`

```
#PBS -S /bin/bash
#PBS -P su58
#PBS -q normal
#PBS -l ncpus=12
#PBS -l walltime=48:00:00
#PBS -l mem=48gb
#PBS -M ana.fabelahinojosa@monash.edu
#PBS -o IsmipF.outlog
#PBS -e IsmipF.errlog
#PBS -l wd

# Source bashrc to set up environment
source $HOME/.bashrc
```

```
# Load needed modules
module load openmpi/4.1.7

# Enable spack
source /home/565/ah3716/spack/0.22/spack-config/spack-enable.bash

# Load spack issm module
spack load issm@ana-local-version-allocation-bugfix %gcc@13

# Run simulation for IsmipF transient solve
# ISSM syntax: TransientSolution <model_dir> <model_name> <execution_dir>
# - <model_dir>: where .bin/.toolkits files are located AND where output
files will be saved
# - <model_name>: base name of model files
# - <execution_dir>: where job executes and temporary files go

mpiexec -np 12 issm.exe TransientSolution /home/565/ah3716/ice_models/S1_F
IsmipF /scratch/su58/ah3716/execution/S1_F
```

no conversions

I copy the whole folder from Gadi containing my `outbin` and also copy my `outlog` and `errorlog` onto my local computer the files are these

```
IsmipF.bin
IsmipF.lock
IsmipF.queue
IsmipF.toolkits
IsmipF.errlog
IsmipF.outlog
IsmipF.outbin <--- I grab this file and process it with `convert_to_nc.py`
```

4. `convert_to_nc.py`

NO `yts` factor so there is no conversion

5. `extract_results.py`

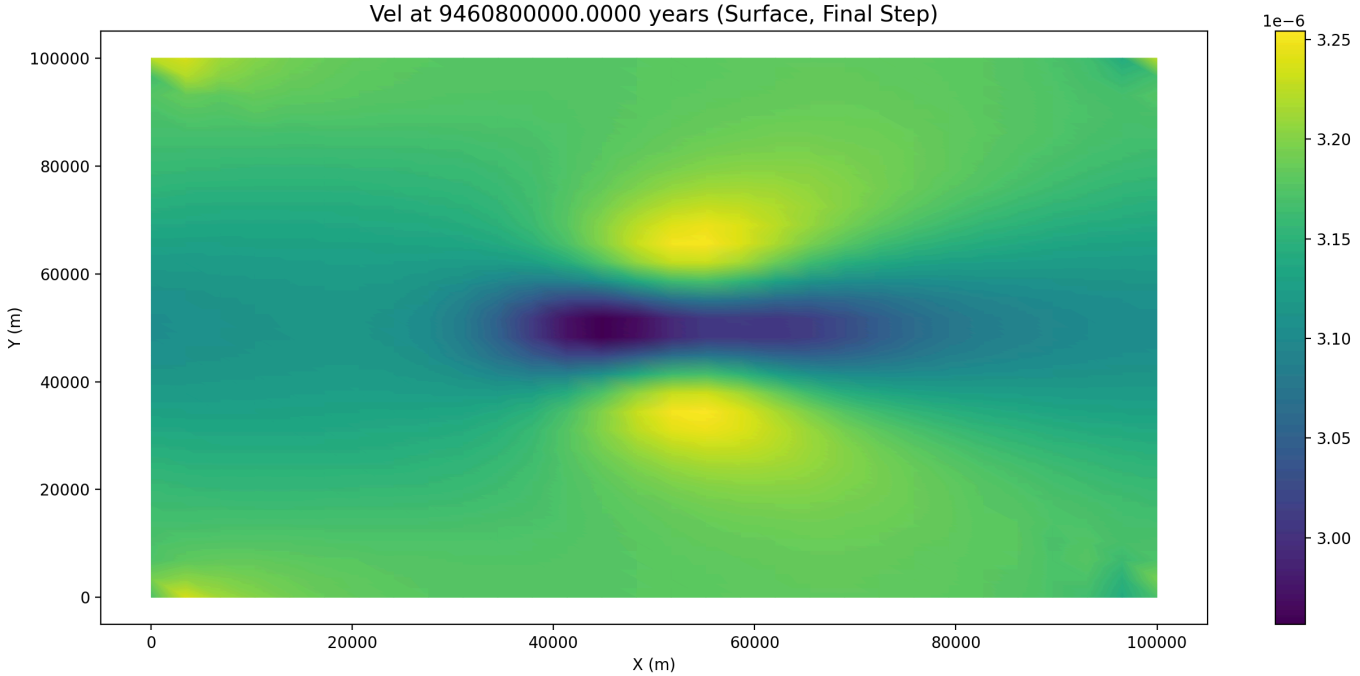
I was assuming that the code was returning velocities in m/a so my `extract_results.py` script . When I extract the raw time data from the nc file:

```
ds = nc.Dataset(results_file, 'r')
tsol_group = ds['results/TransientSolution']
```

```
times = tsol_group.variables['time'][:]
```

The titles of y plot are obviously wrong

GADI output

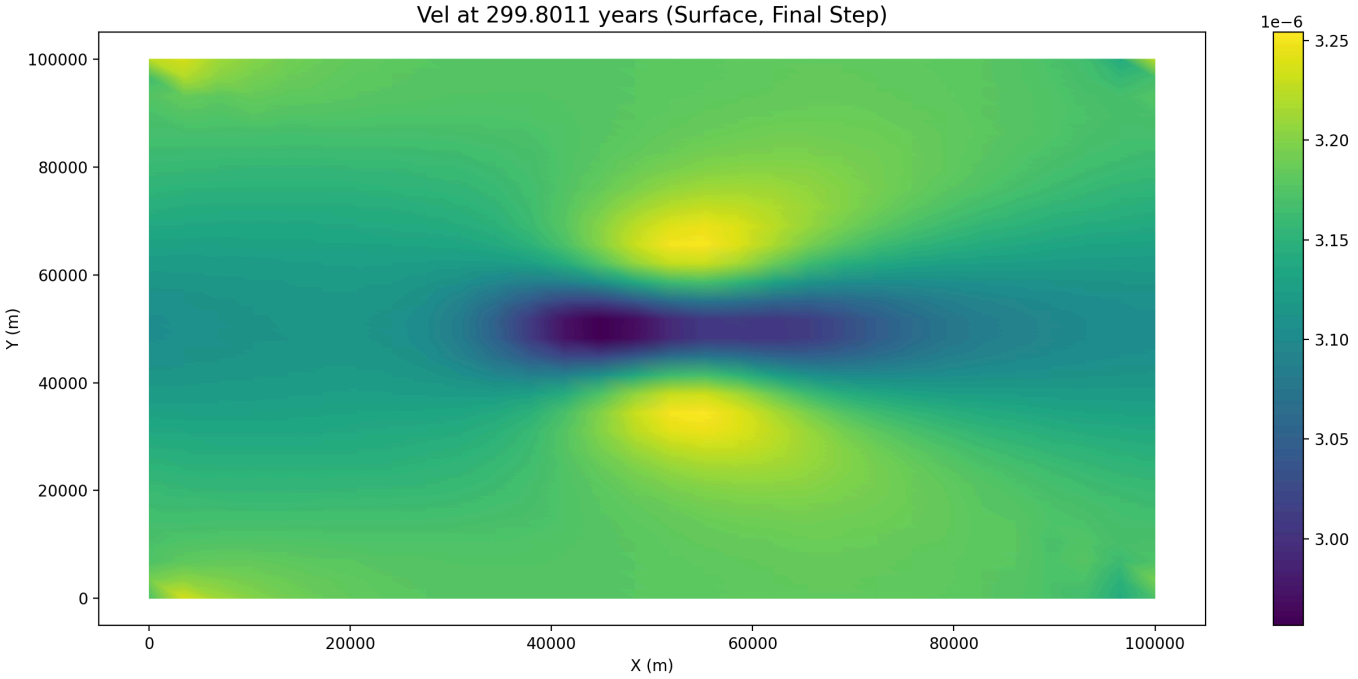


(the super low velocities were (to me) not an issue I had power over... until you guys told me so yesterday). To to solve the obviously wrong time in the title I did have this conversion in my `extract_results.py`

```
SECONDS_PER_YEAR = 31556926.0
...
times_in_years = tsol_group.variables['time'][:] / SECONDS_PER_YEAR
```

Ta-dah! The out put plots titles get fixed!

GADI output

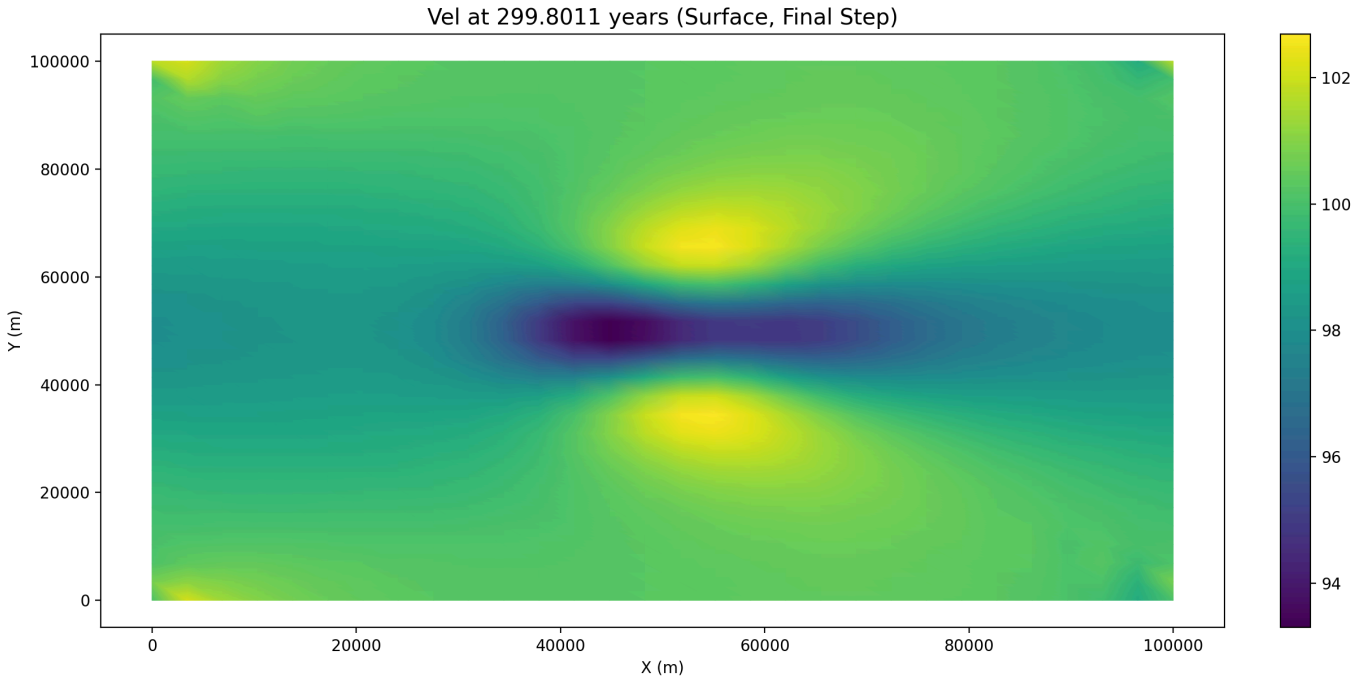


I hope that makes sense.

Anyway, after we noticed that I had the plot velocities were wrong by a factor of `yts`
I converted the velocities too.

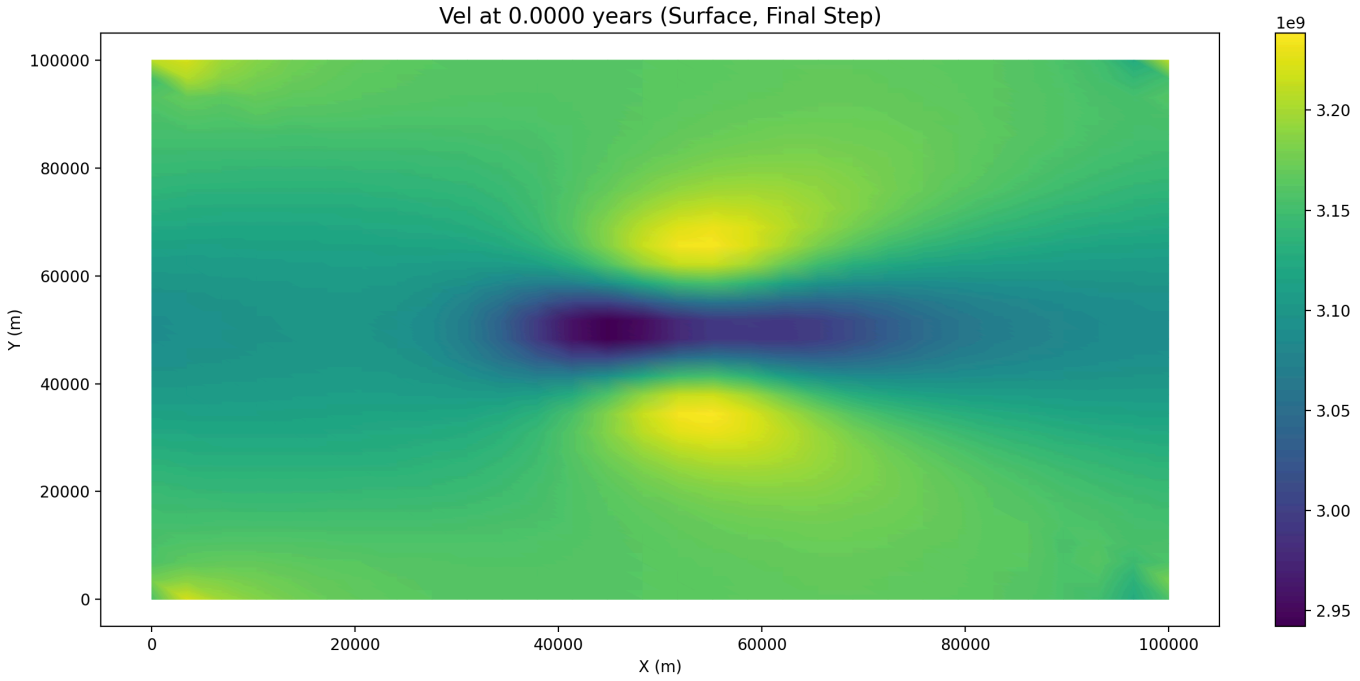
```
SECONDS_PER_YEAR = 31556926.0
...
times_in_seconds = tsol_group.variables['time'][:]
times_in_years = times_in_seconds / SECONDS_PER_YEAR

if field_name in ['Vx', 'Vy', 'Vz', 'Vel']:
    data_for_step = data_for_step * SECONDS_PER_YEAR

max_velocities = np.max(vel_data * SECONDS_PER_YEAR, axis=1)
```

That combination of conversions returns the correct values for the velocities and time in my plots.
GADI output



Which is all very nice... but then I thought it was weird that I needed to convert the units of velocity because I remembered that ISSM should return velocity units in m/a in the final output. So I though I should run a local simulation with everything identical to what I ran on Gadi and try to use my newly improved version of `extract_results.py` to see what happened.

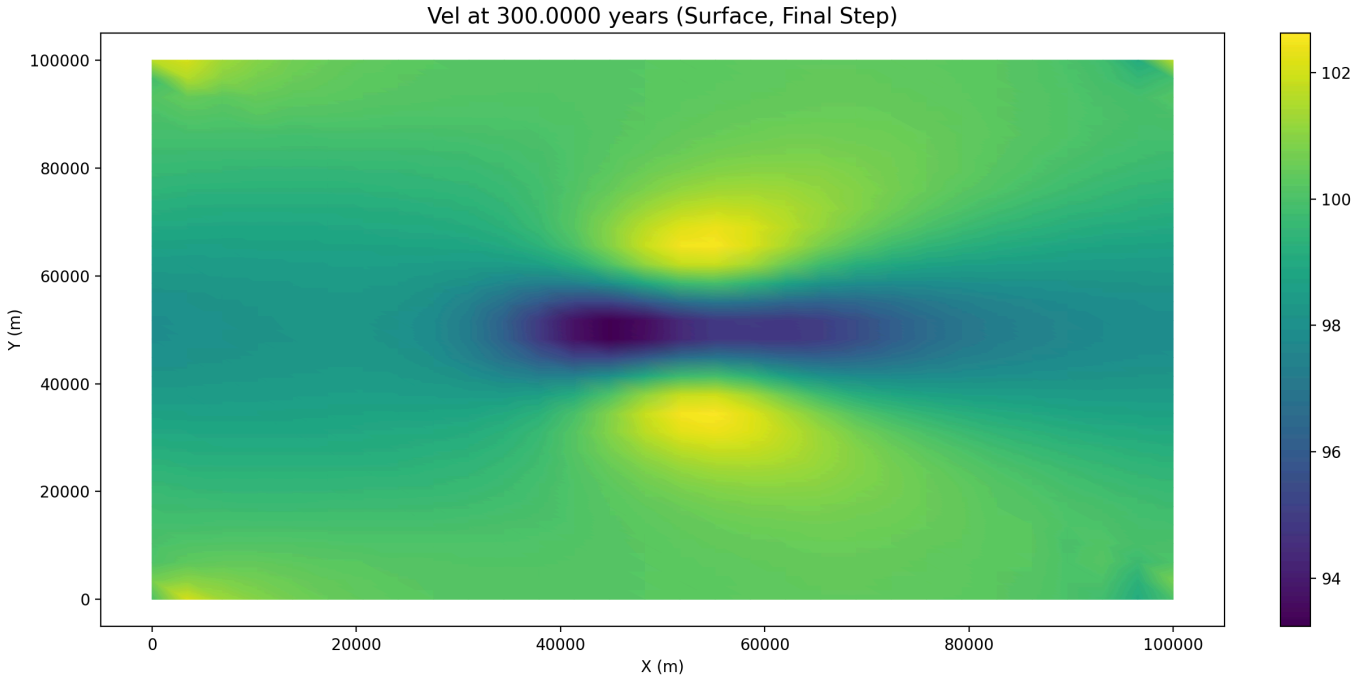
LOCAL output



As you can see the time is messed up (this should be ~300 years but by converting it I make it too tiny to be displayed) and the velocity is messed up too... 🤦‍♀️

I have not identified any place in my workflow that converts the units other than that in `extract_results.py` as I already described. Reverting the units conversion for the velocity when using a local simulation returns the correct plots

LOCAL output



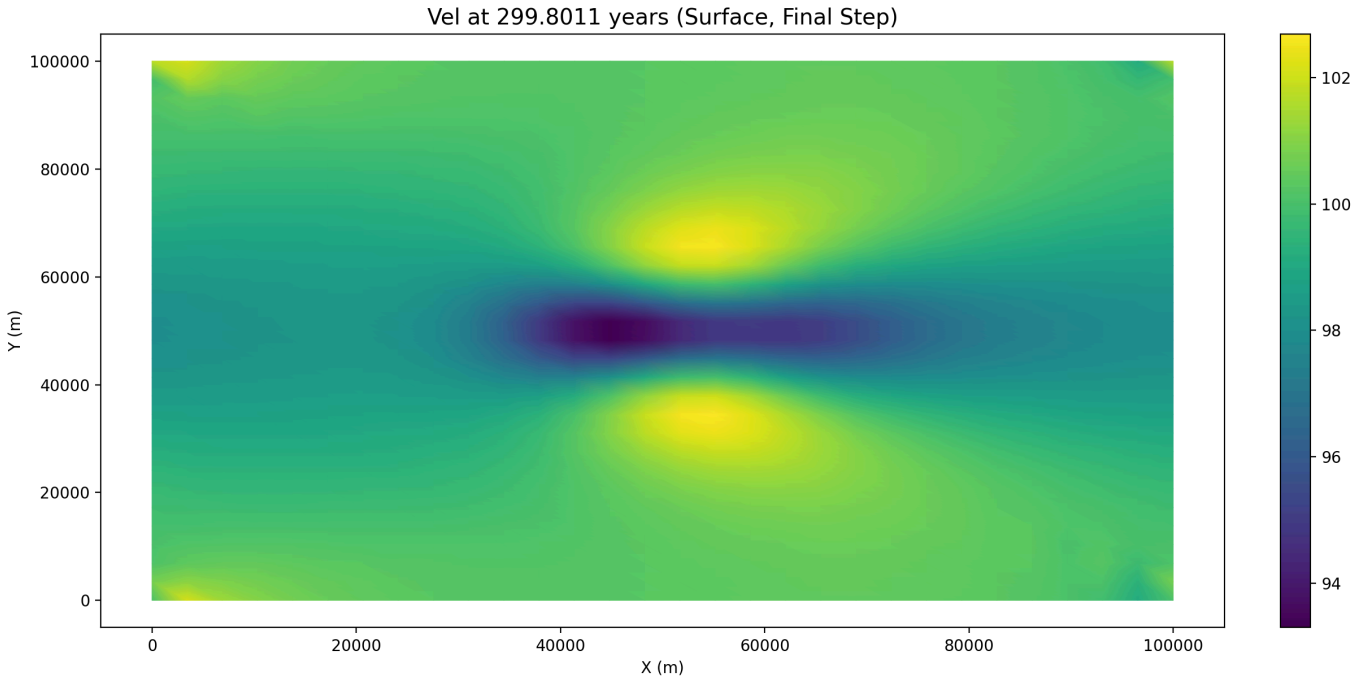
Something that you might have noticed too is that the plots for GADI output and the LOCAL output have a different final time by 0.2 years... I don't know why that is.

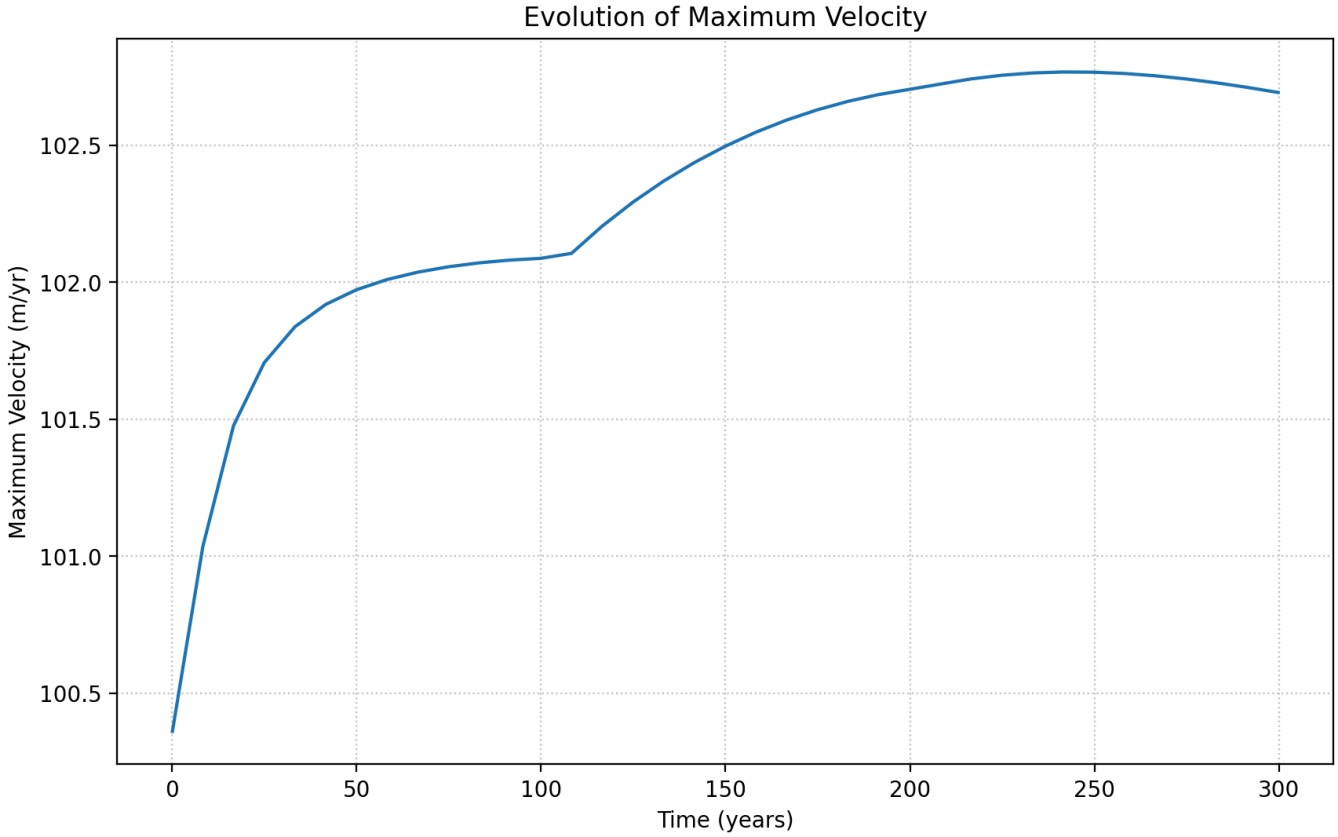
TRYING OUT RUNNING LOCAL LIKE ON GADI

intead of using python I ran issm.exe locally similar to gadi `queue`

```
$ISSM_DIR/bin/issm.exe TransientSolution
/home/ana/Desktop/code/ISMIP/processing IsmipF
/home/ana/Desktop/code/ISMIP/processing
```

SO IF I convert this `.outbin` using my `convert_to_nc.py` and extract the data with the Gadi version of `extract_*.py` (the script that converts time and velocities)





I get the correct units!

Maybe what is happening is that the `outbin` file doesn't have undergo a step to convert from ISSM internal units (s and m/s) into the output units (yrs and m/a)

■