for_wintertime_users

April 17, 2023

1 NetCDF Tutorial

Maxime Benoît-Gagné

This tutorial was given at the meeting of the Numerical Ecology in Oceanography Laboratory (NEOLab), Université Laval, on April 21, 2023.

This Jupyter notebook is part of the project for_wintertime_users containing documentation and examples on how to use the outputs of the default simulation EXP-0 described in Benoît-Gagné et al. (submitted).

The documentation and examples are available on:

https://github.com/maximebenoitgagne/for_wintertime_users/tree/main.

The original project of Benoît-Gagné et al. (submitted) can be found on:

https://github.com/maximebenoitgagne/wintertime/tree/v1.5.

2 Definitions

NetCDF file for a region: A NetCDF file for many latitudes and many longitudes, for example, a NetCDF file for the whole Baffin Bay.

NetCDF file for a specific location: A NetCDF file for only one latitude and one longitude, for example, a NetCDF file for the location of an ice camp.

Results of Benoît-Gagné et al. (submitted): NetCDF files containing the results of the default simulation of Benoît-Gagné et al. (submitted). The simulation was adapted to the location of the Qikiqtarjuaq sea ice camp in western Baffin Bay (67.4797°N, -63.7895°E) in 2016 during the Green Edge sea ice camp mission. The model was run with a spinup of 10 years to stabilise the system. Use the data of that tenth year. The dimensions of the data are the depth steps and the time steps.

3 What you will learn in this tutorial

- Find the results of Benoît-Gagné et al. (submitted).
- Understand the structure of the results of Benoît-Gagné et al. (submitted).
- Explore the structure of a NetCDF file for a specific location with Panoply.
- Plot a NetCDF file for a specific location with Panoply.

- Read a NetCDF file for a specific location with Python.*
- Export a NetCDF file for a specific location into a comma-separated values (CSV) file with Python.*
- Plot a variable from the results of Benoît-Gagné et al. (submitted) with Python.*
- Find more examples of Python scripts reading the results of Benoît-Gagné et al. (submitted).*

Note: The goals with an asterisk (*) have the following prerequisites: * Knowledge of Anaconda. * Basic knowledge of Python.

4 What you will not learn in this tutorial

- Anaconda.
- Python.
- Use a NetCDF file for a specific location when the metadata doesn't contain sufficient information to retrieve the depth steps and the time steps of the file. Actually, the only solution would be to ask someone who knows the answer, for example, the human author of the files.
- Export a NetCDF file for a specific location into a CSV file with Panoply.
- Write a NetCDF file.
- Read a NetCDF file for a specific location in R.
- Use a NetCDF file for a region.

5 Installation steps

These installation steps were tested for MacOS X 10.13 (High Sierra).

- Install Panoply (https://www.giss.nasa.gov/tools/panoply/download/).
- Install Anaconda for Python 3 as described on https://datacarpentry.org/python-ecology-lesson/setup.html:
  - Go to https://www.anaconda.com/products/distribution.
  - Click Download.
  - Double-click the graphical installer `.pkg` file.
  - Follow the instructions. Select the default settings.
  - Verify the installation by entering the following command in the Terminal (Launchpad icon in the Dock, type Terminal, click Terminal):

  ```
  conda --help
  ```

- Enter the following commands in the Terminal. The creation of the conda environment can take approximately 15 minutes.

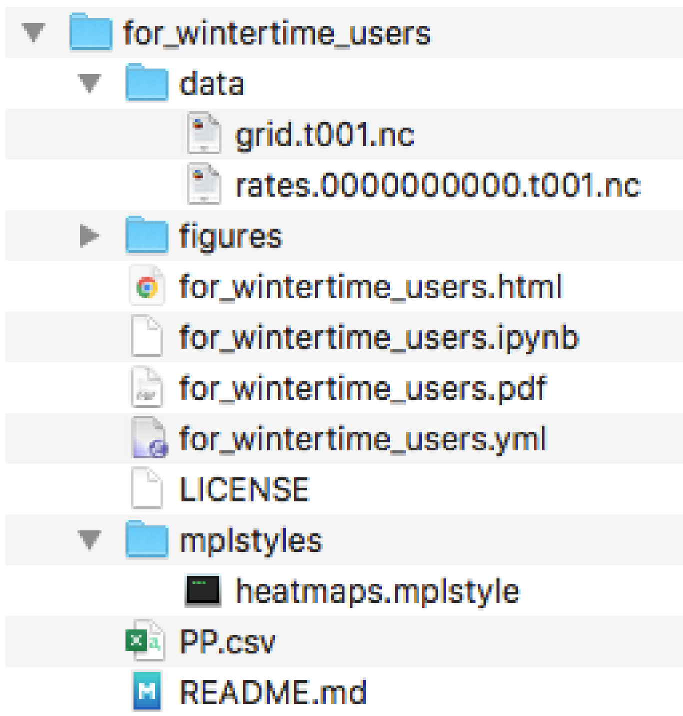
```
git clone git@github.com:maximebenoitgagne/for_wintertime_users.git
cd for_wintertime_users/
conda env create -f for_wintertime_users.yml
conda activate for_wintertime_users
jupyter notebook
```

Once the environment for_wintertime_users is created, the Jupyter notebook can be launched simply with:

```
cd for_wintertime_users/
conda activate for_wintertime_users
jupyter notebook
```

6 Find the data

The results of Benoît-Gagné et al. (submitted) are available on https://github.com/maximebenoitgagne/wintertime/tree/v1.5/data/DataS8_output_mitgcm/exp0. Two relatively small files of these results are also in the directory `data` of this project.

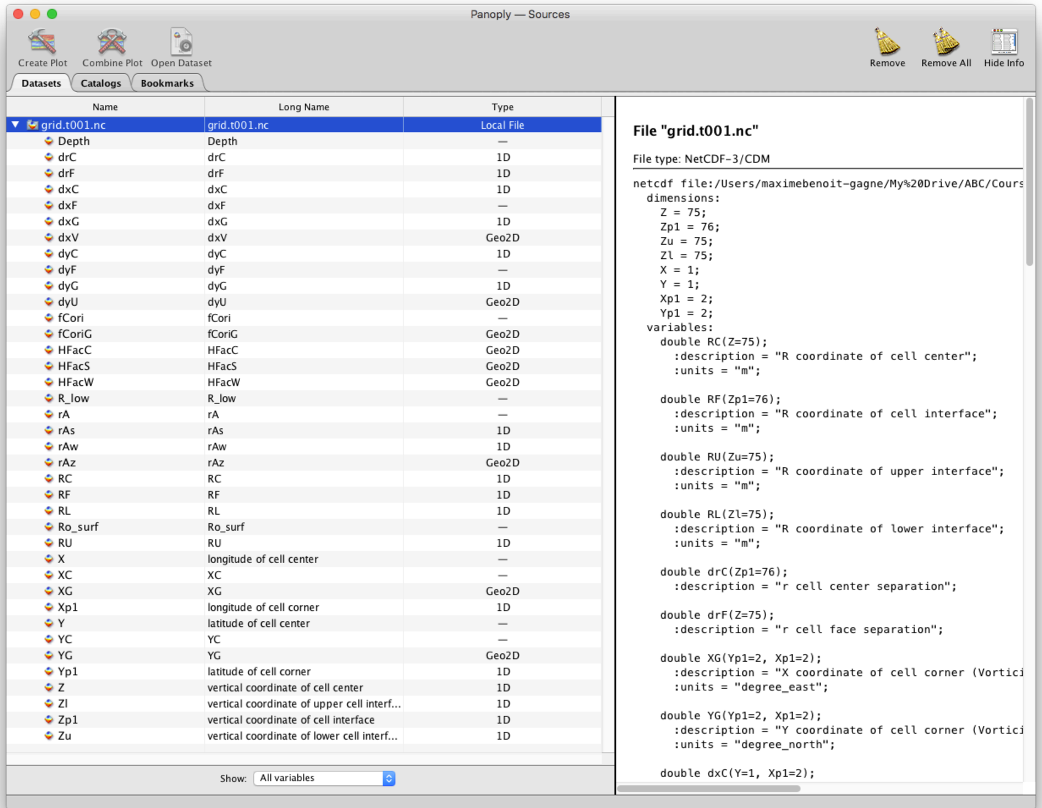


7 Read a NetCDF file with Panoply

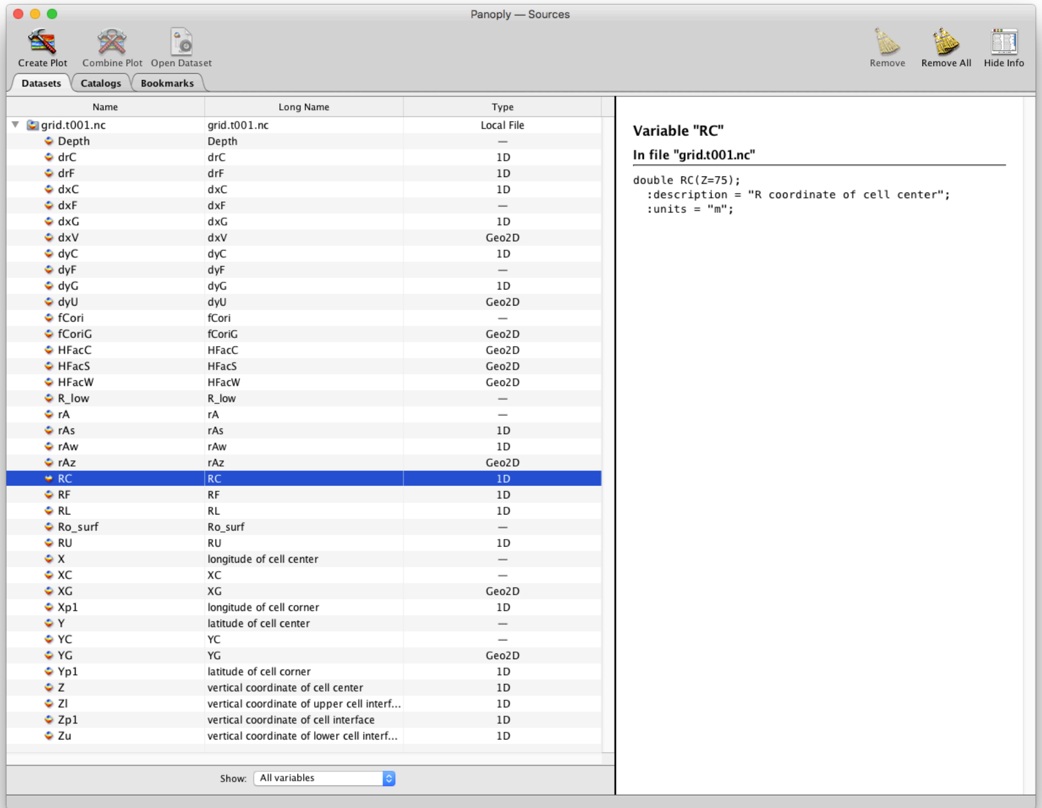
Panoply is a graphical user interface (GUI) tool to explore and plot NetCDF files.

Steps

- Click on Panoply.
- Select grid.t001.nc.

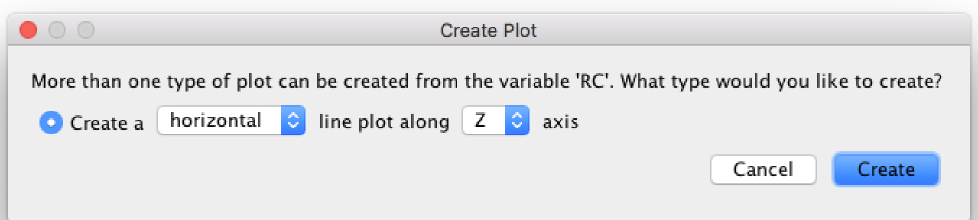


- Select the variable RC.

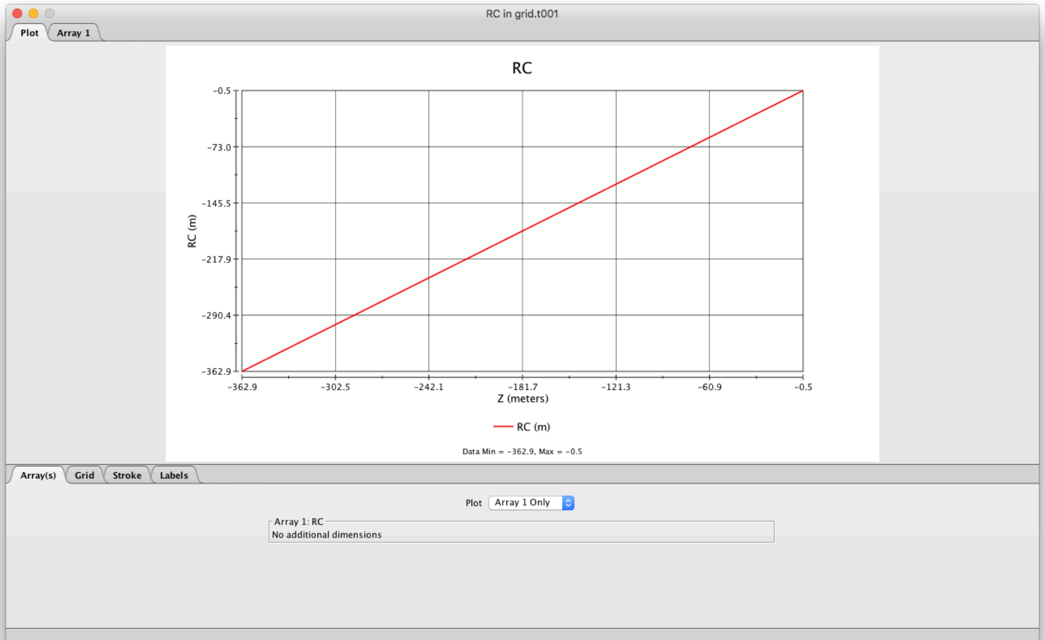


We see that the variable RC in grid.t001.nc contains the values of the 75 depth steps.

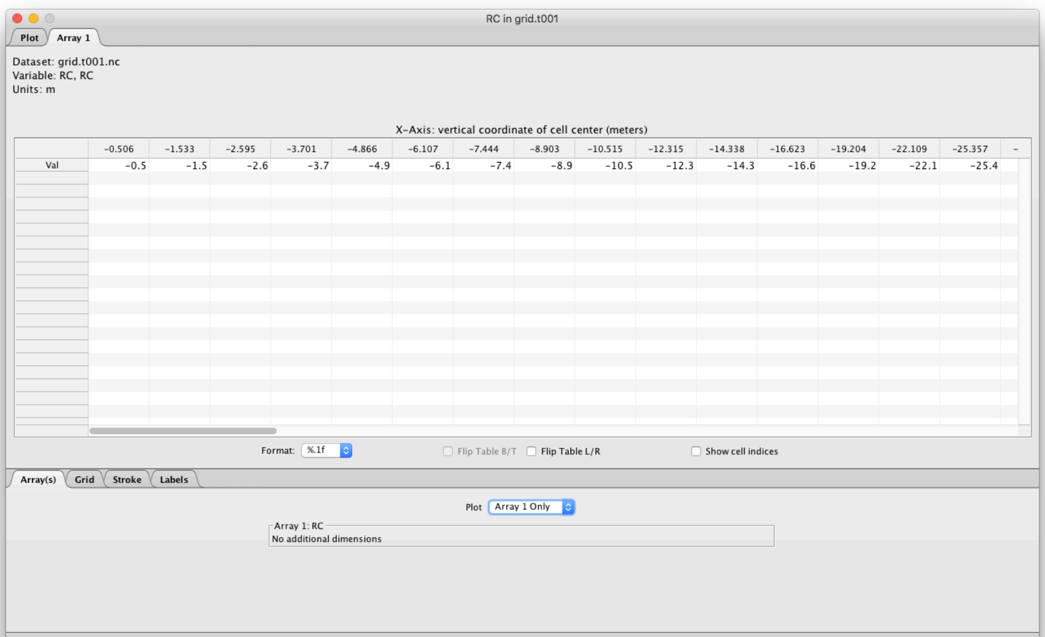
- Double-click on RC.



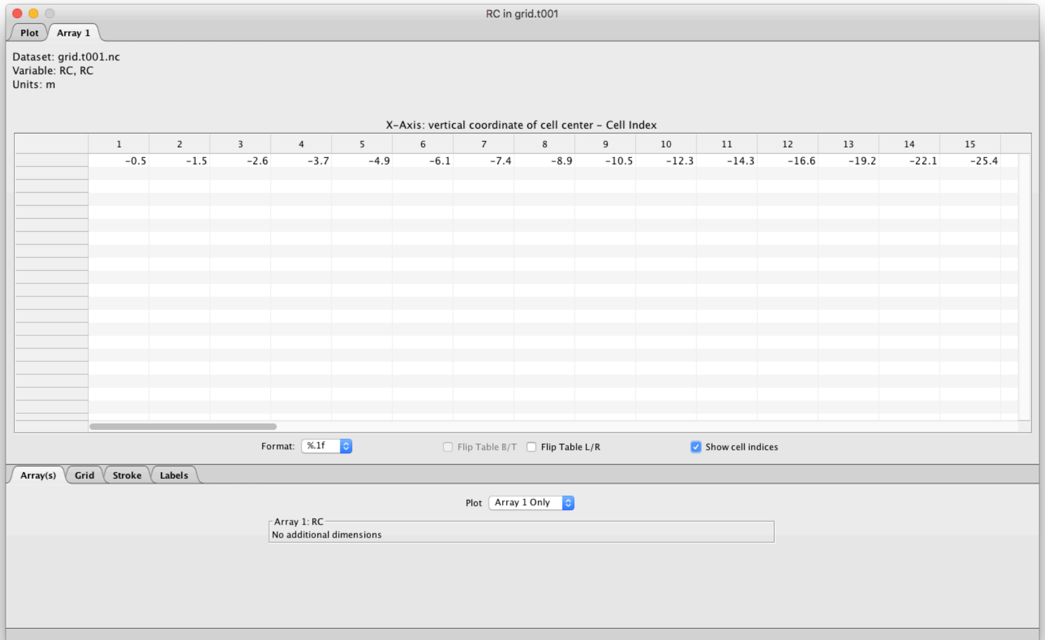
- Click Create.



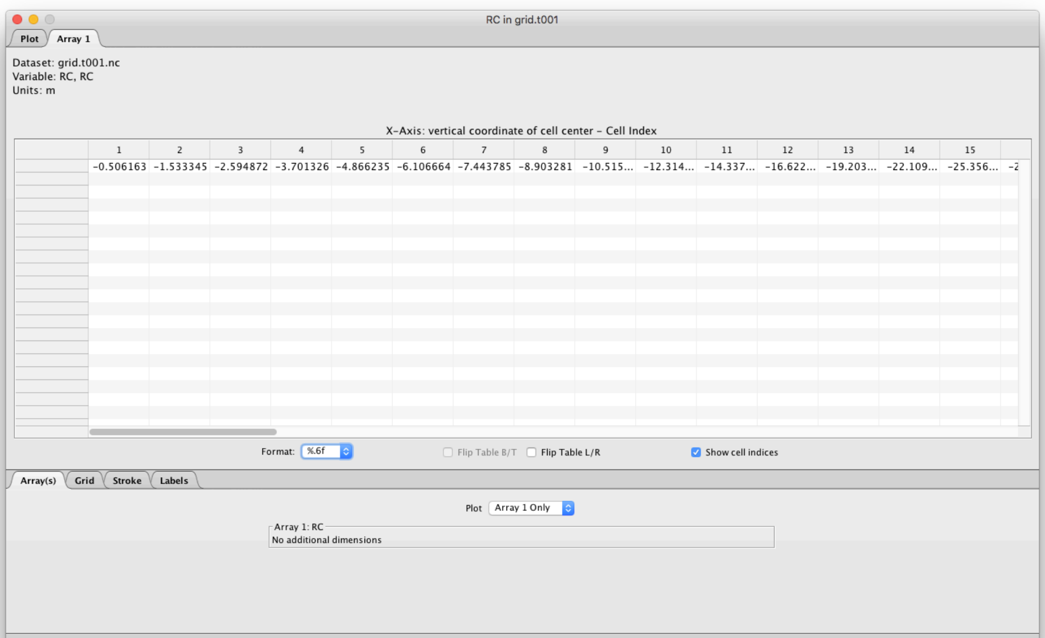
- Select tab Array 1.



- Check Show cell indices.



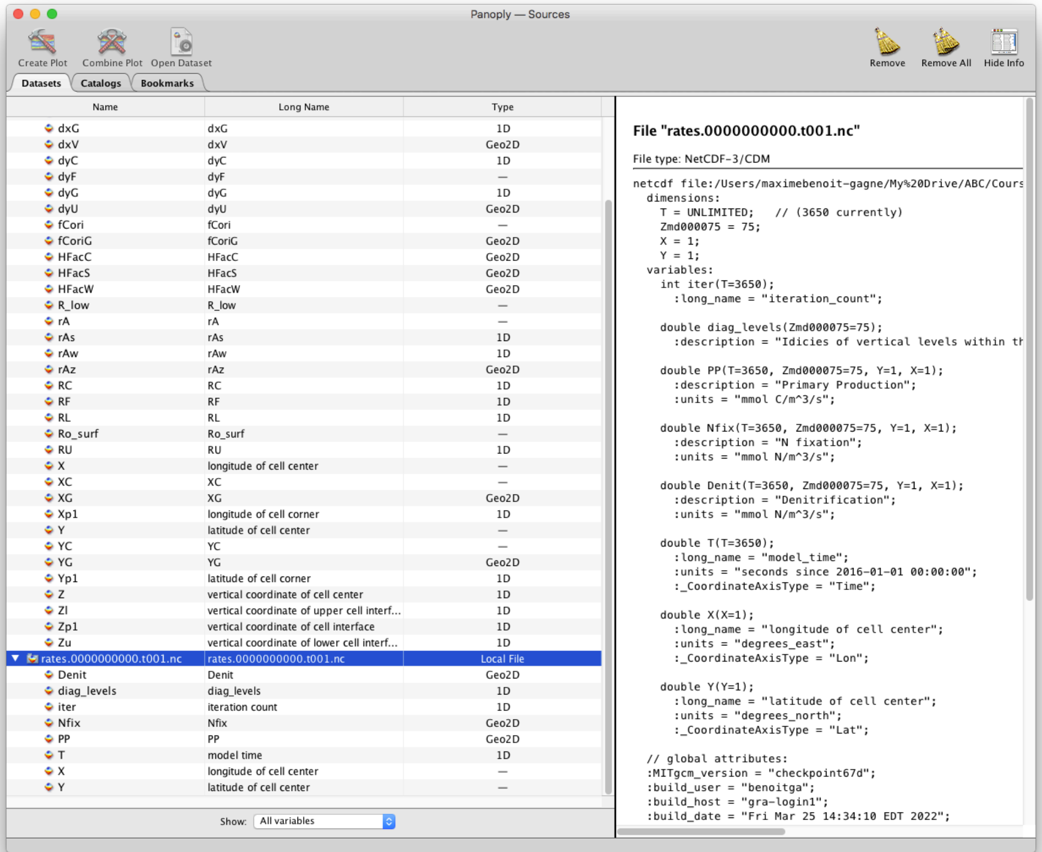
- Select Format %.6f.



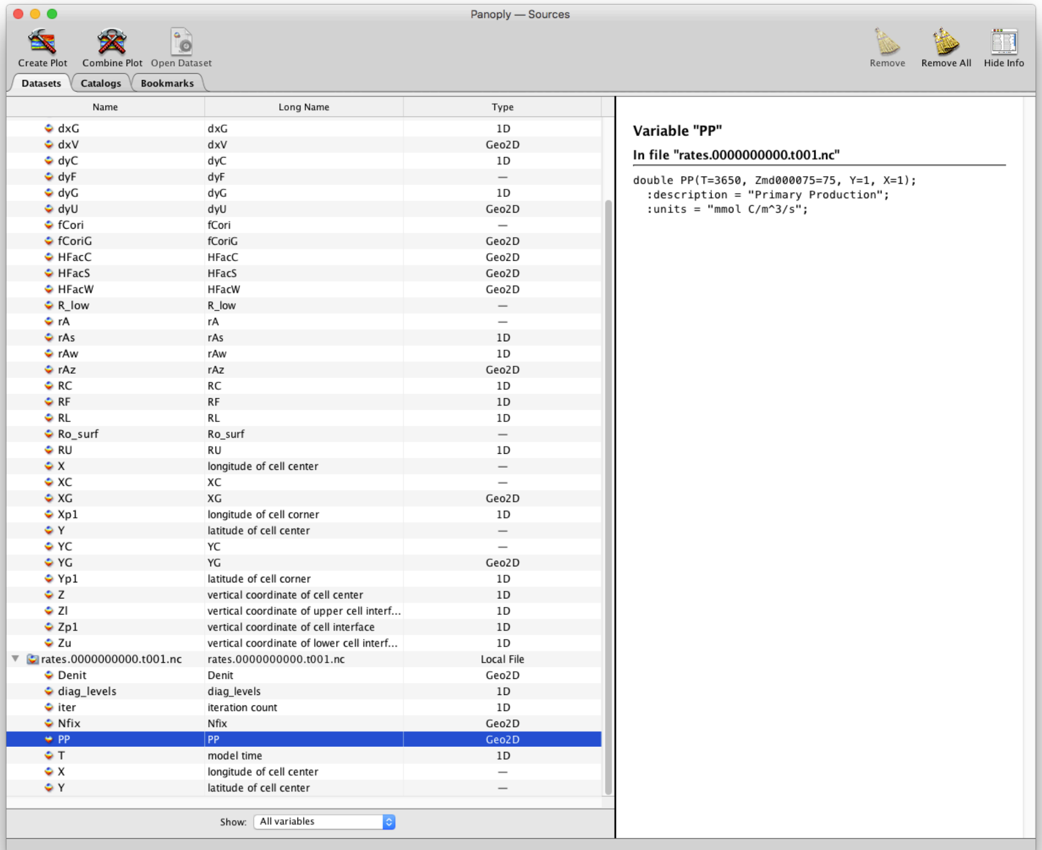
We now see explicitely the values of the 75 depth steps. Note that the vertical grid is irregular. We have to take that into account if we want to transform units by m^{-3} into units by m^{-2}.

We will now open another NetCDF file. We will open rates.0000000000.t001.nc.

- File > Open > rates.0000000000.t001.nc.



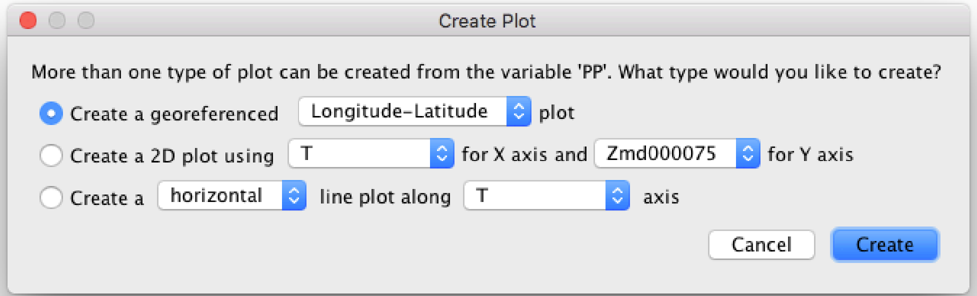
- Select the variable PP.



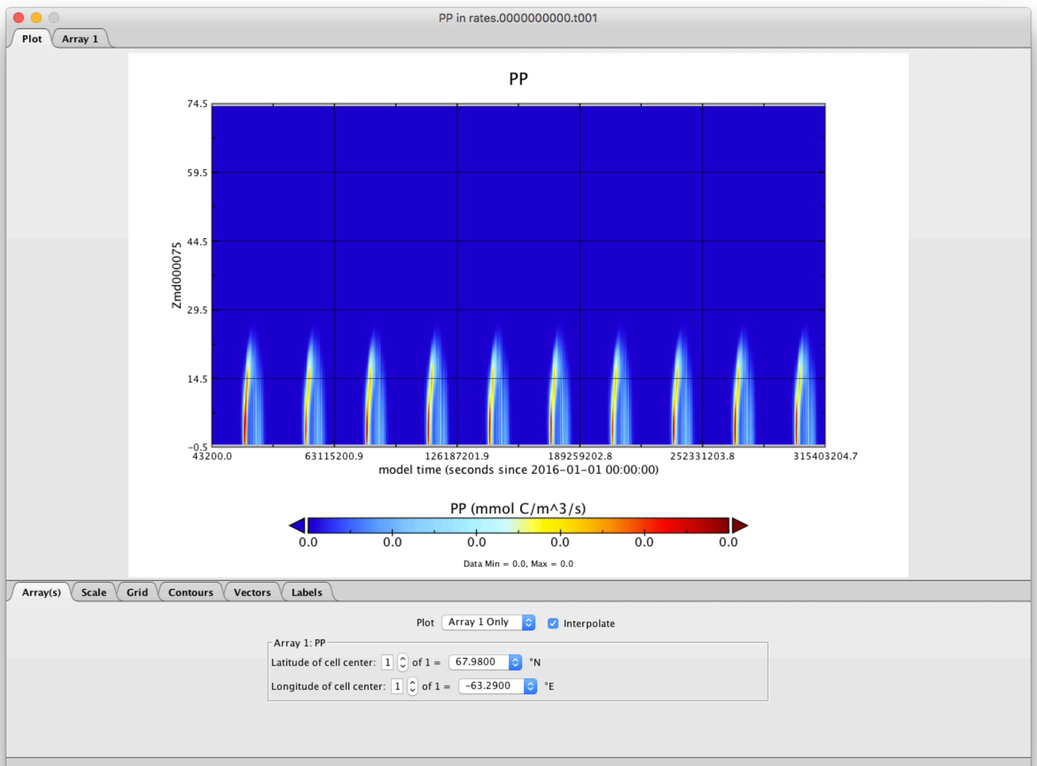
We see that the variable PP is the primary production in mmol C m^{-3} s^{-1}. The dimension T is the daily time steps over 10 years (10*365=3650). The dimension Zmd000075 is the 75 depth steps.

8 Plot a NetCDF file with Panoply

- Double-click on PP.



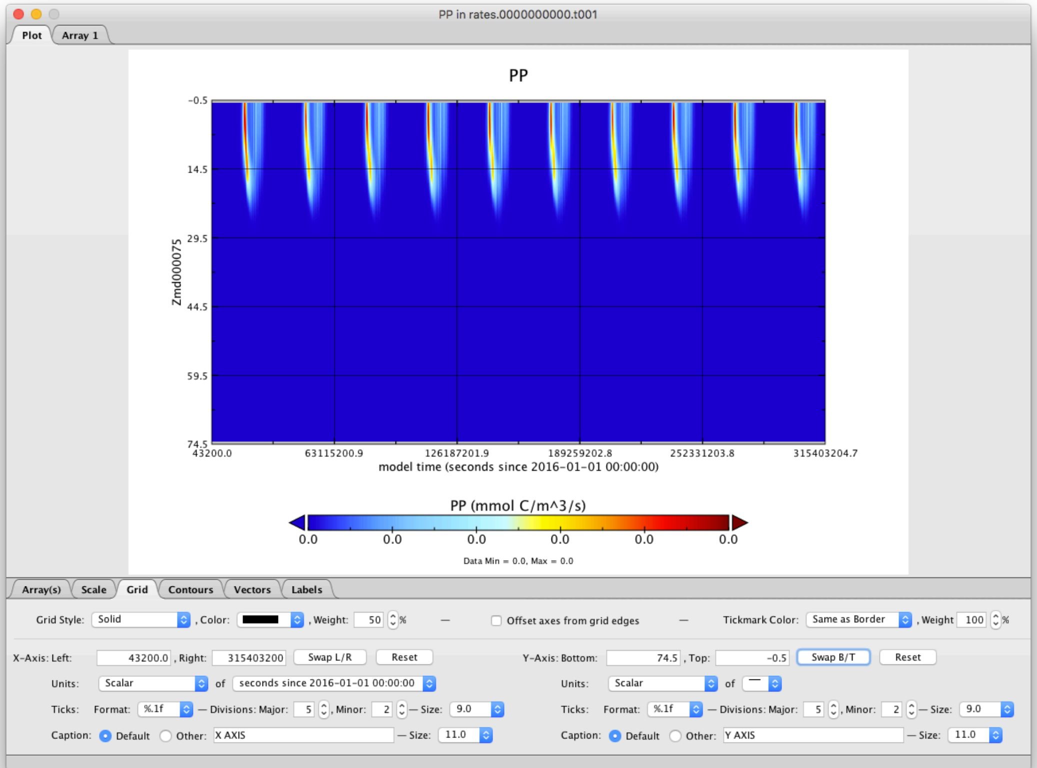
- Check Create a 2D plot using T for X axis and Zmd000075 for Y axis.
- Click Create.



Note that the Y axis contains the indices of the depths and not the values of the depths. The values of the depths are not in the file rates.0000000000.nc. They are only in the variable RC of grid.t001.nc.

Note also that the depths on the Y axis are in the reverse order. This is because the first depth index is at the water surface and is shown at the bottom of the plot. We would rather show it at the top of the plot.

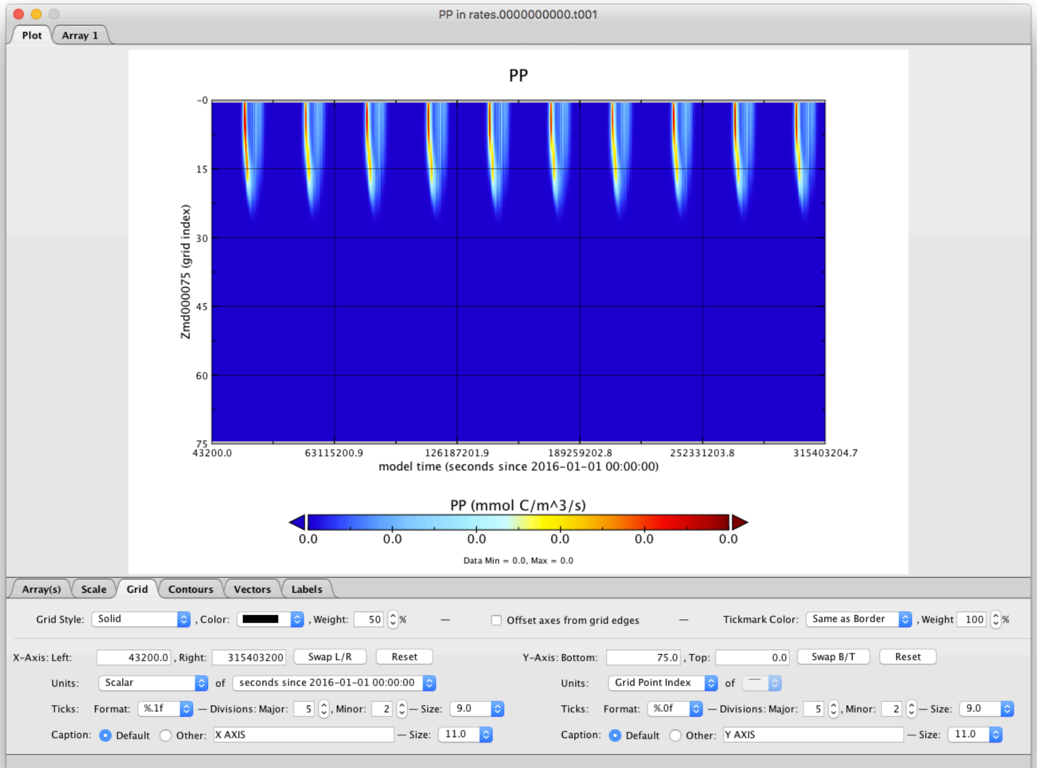
- Grid > Swap B/T.



We now see the primary productivity over 10 years. We can see the 10 yearly spring blooms. It worths noting again that the Y axis contains the indices of the depths from 0 to 74 and not the values of the depths.

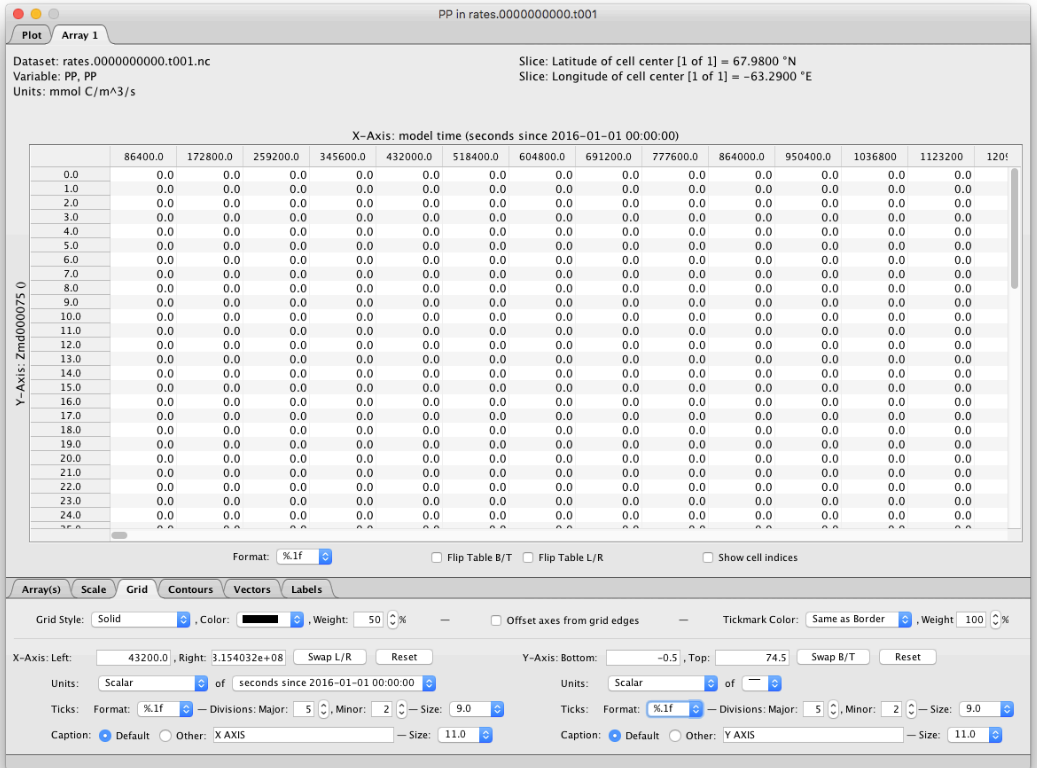
To make it clearer, we will show the indices of the depths from 1 to 75 instead of 0 to 74. In order to do that:

- Y-Axis: Units: Grid Point Index.
- Y-Axis: Ticks: Format %.0f.
- Swap B/T.

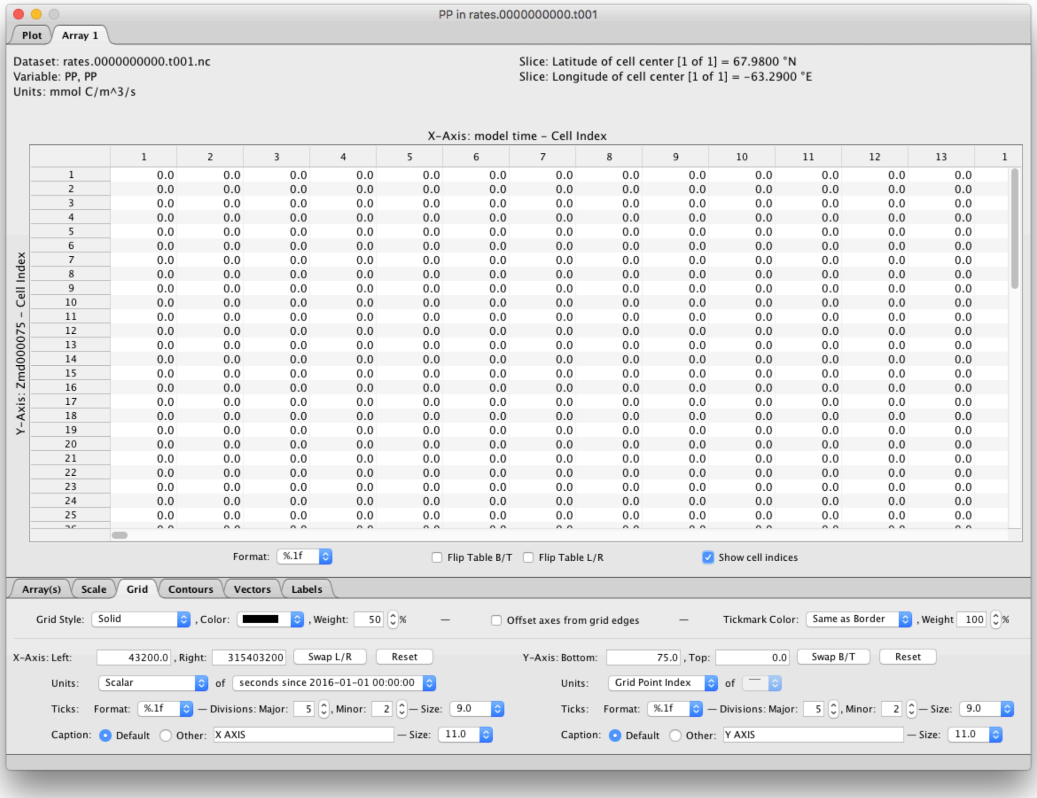


Verify.

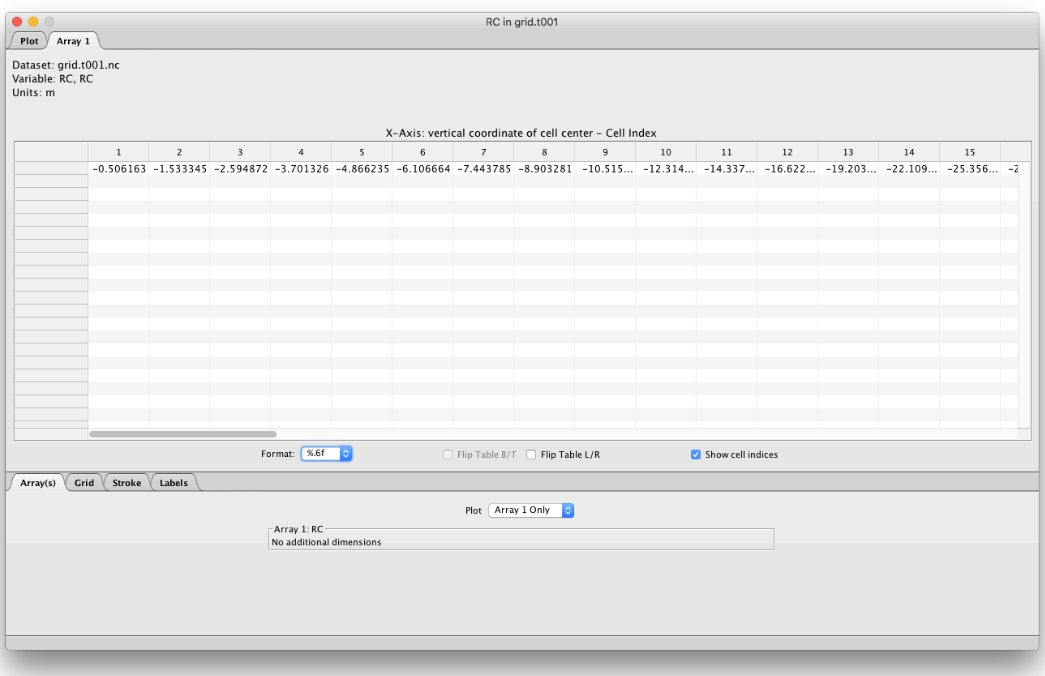
- Select tab Array 1.



- Show cell indices.

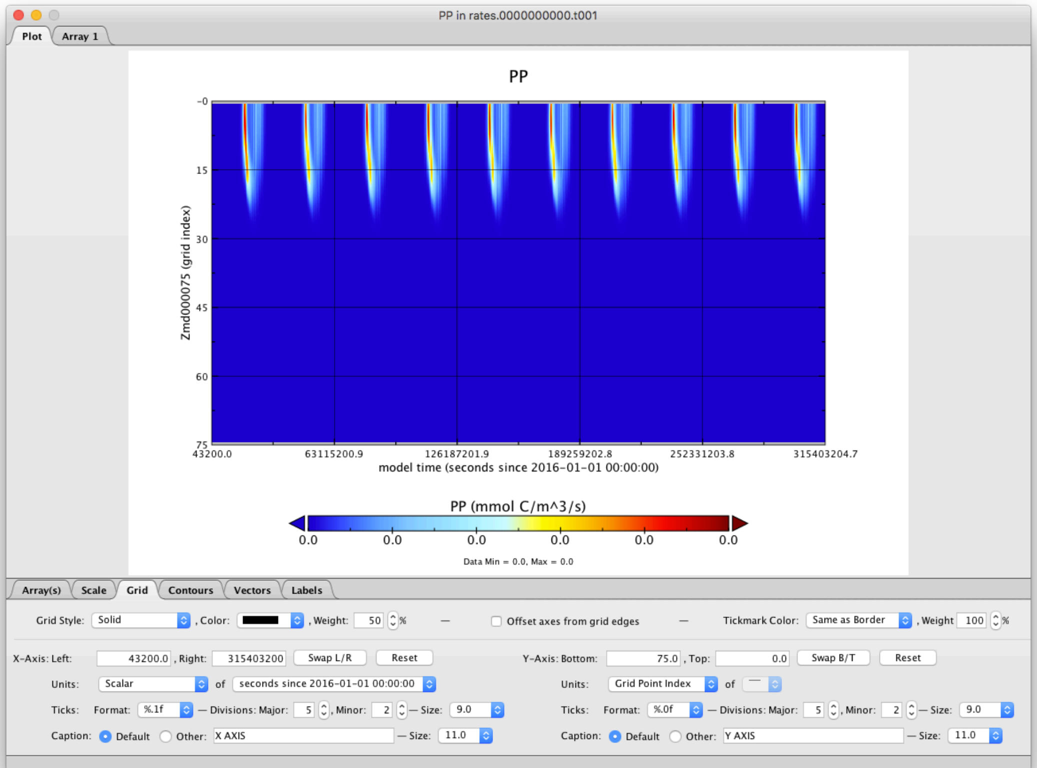


We will retrieve the value of the depth at the depth index 15 using the variable RC in grid.t001.nc.



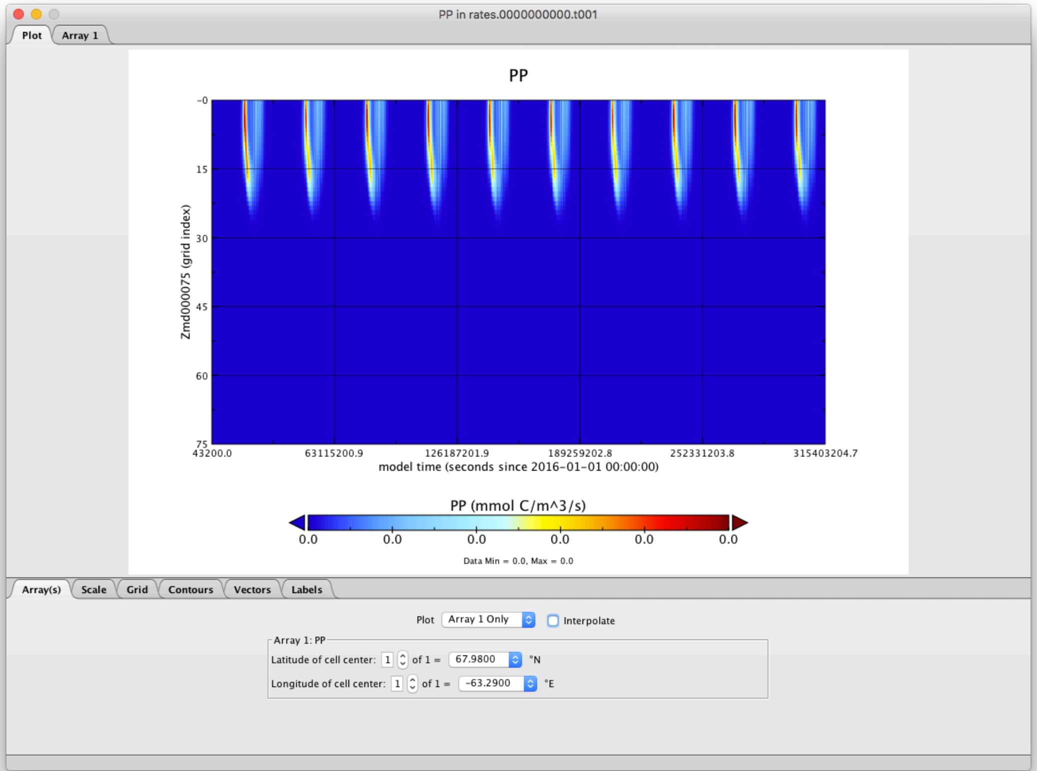
We see that the value of the depth at the depth index 15 is 25.356 m.

- Select the tab Plot.



To make the rows corresponding to each depth clearer:

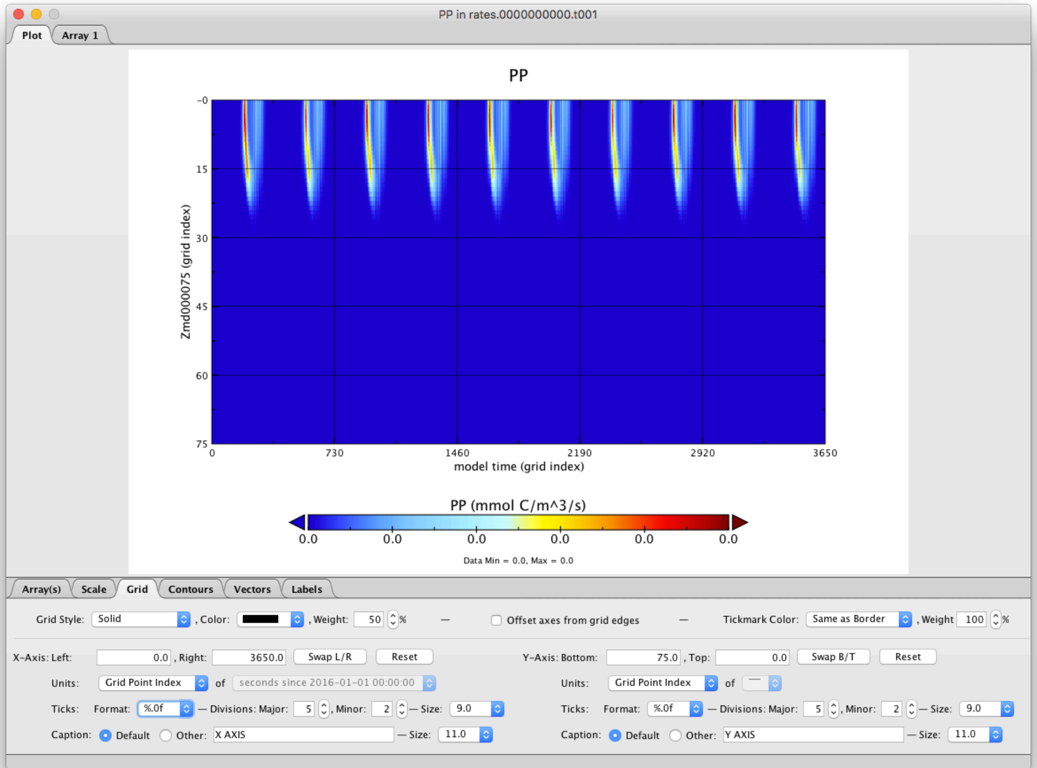
- Select the tab Array(s).
- Uncheck Interpolate.



The fifteenth row, just above the tick 15 on the Y-Axis, corresponds to a depth of 25.356 m.

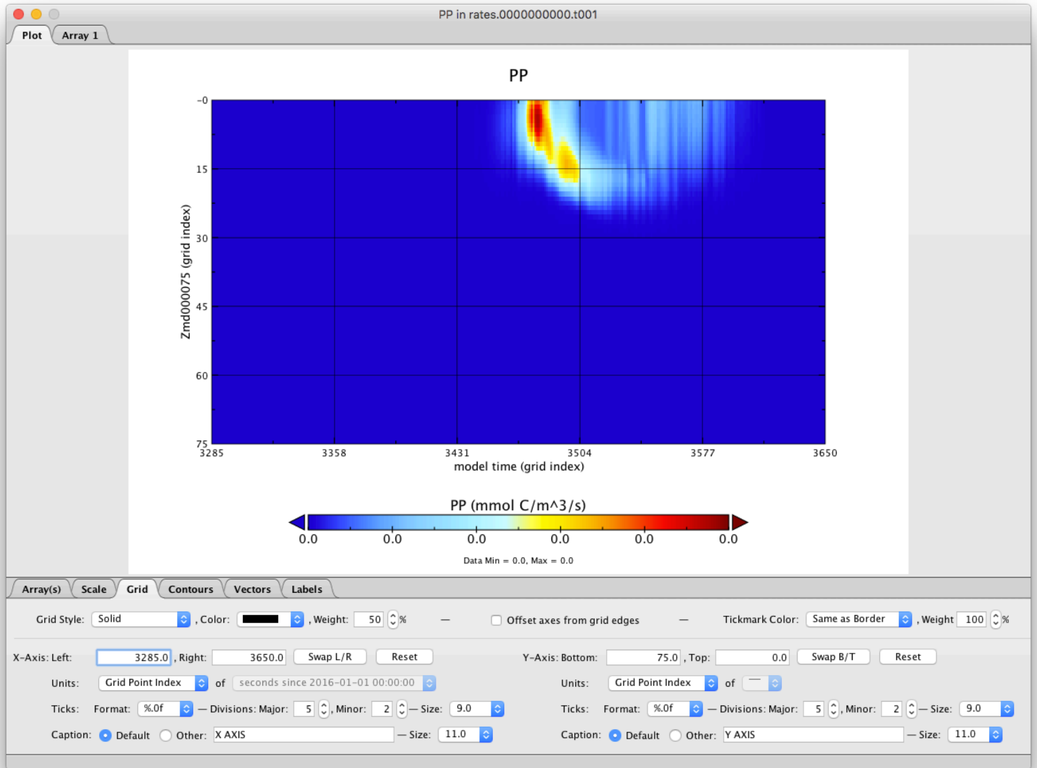
We will now make the X-Axis clearer by showing the indices of the time steps.

- Select the tab Grid.
- X-Axis: Units: Grid Point Index.
- X-Axis: Ticks: Format: %.0f.



The X axis is now the index of the day over the 10 years of the spinup. We want to select only the tenth year. The indices of the X axis will be 365*9=3285 (inclusive) to 365*10=3650 (exclusive).

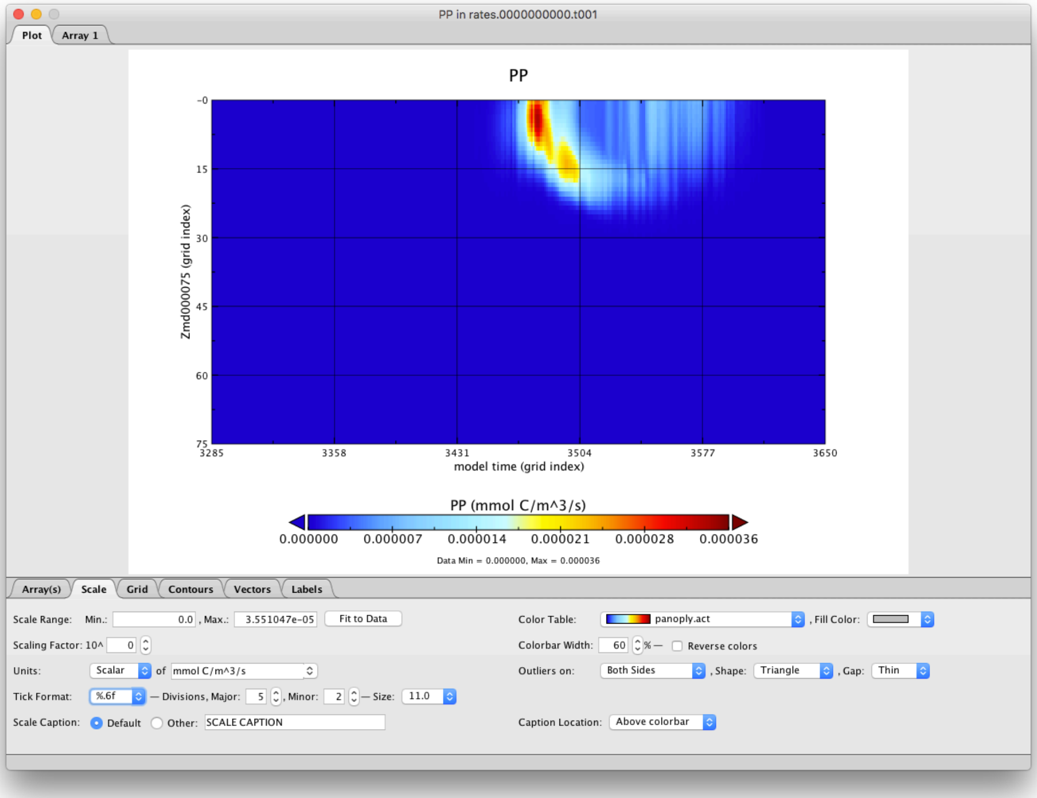
- X-Axis: Left: 3285.



It worths noting that the selection of the tenth year in the Grid tab affects only the plot in the tab Plot. It doesn't affect the array in the tab Array 1. The array in the tab Array 1 will continue to contain the values for the 10 years.

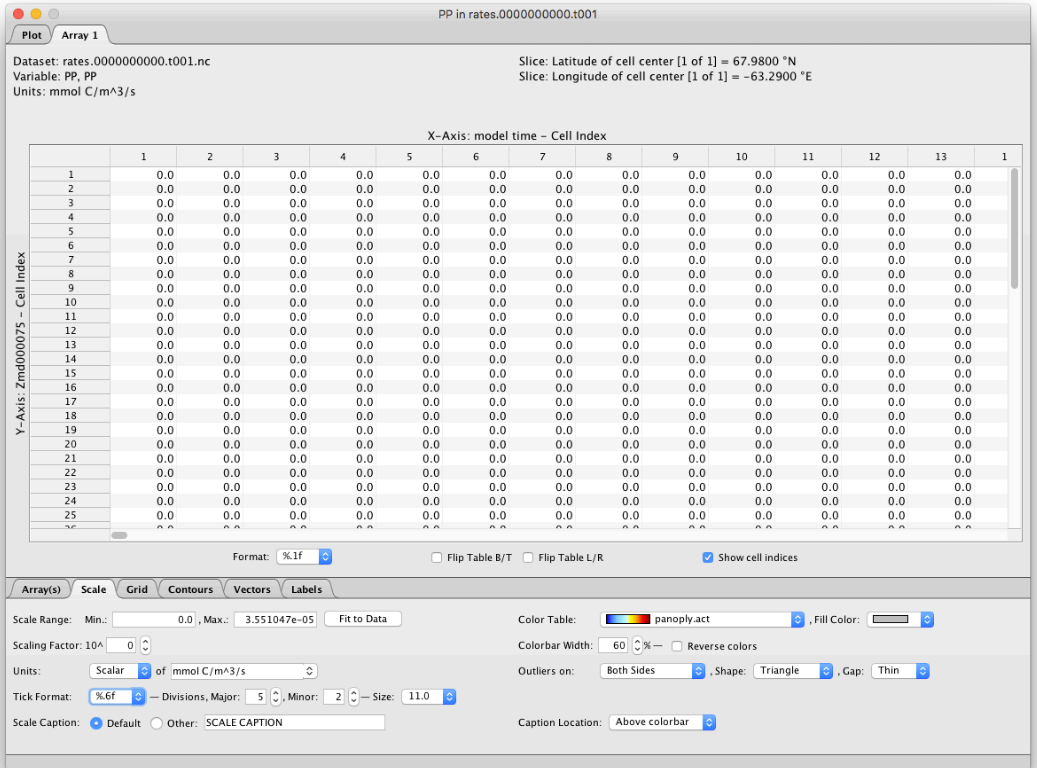
We can improve the precision of the colour bar:

- Select tab Scale.
- Tick Format: %.6f.



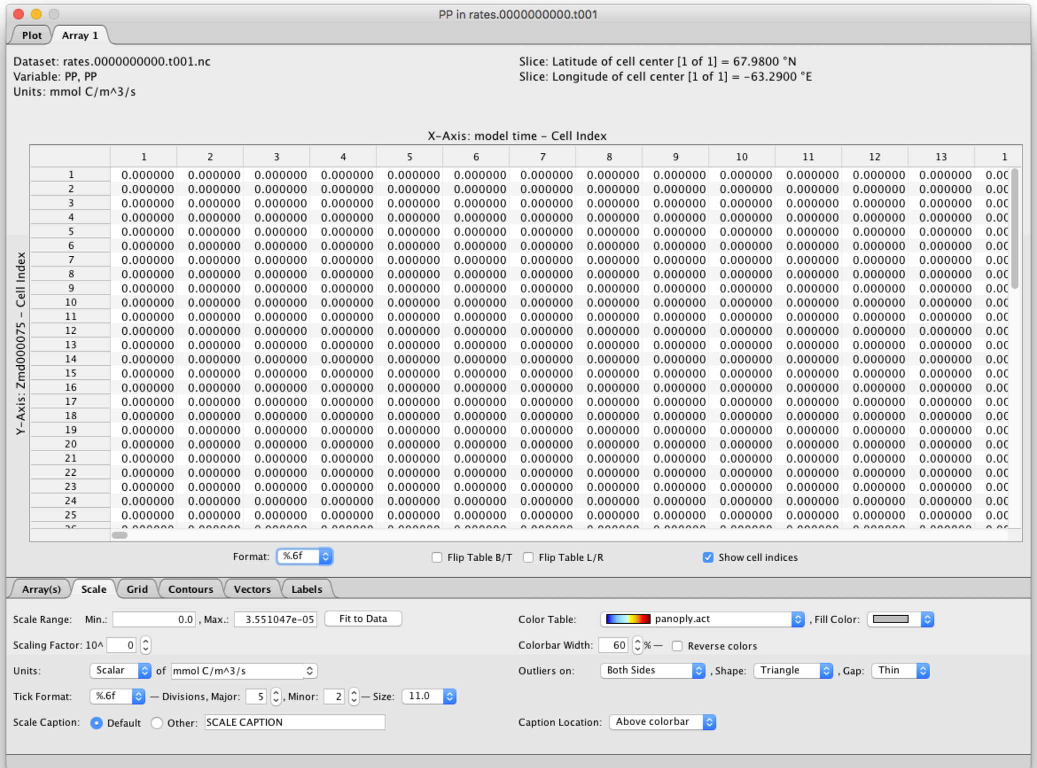
We can find a specific numeric value in the tab Array 1.

- Select tab Array 1.



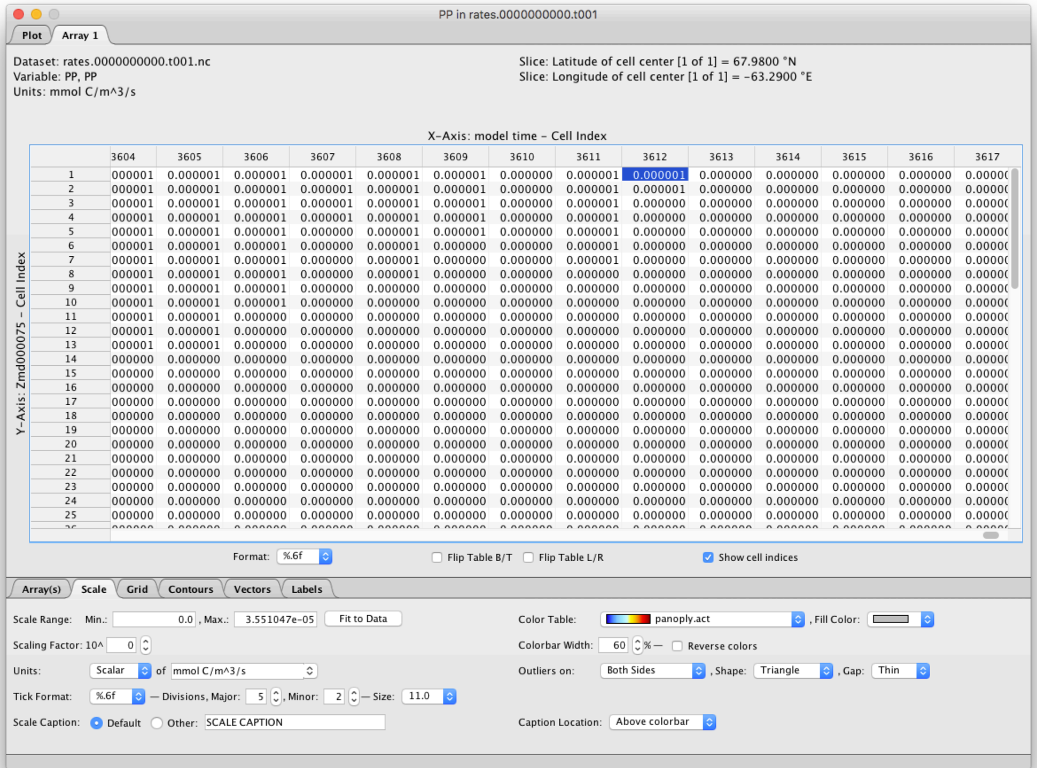
We can improve the precision.

- Format: %.6f.



We now see explicitely the values of the 75*3650 primary productivities.

Move the horizontal scroll bar to the right to see some values above 0.



9 Read a NetCDF file with Python

9.1 First, load libraries

[1]:
```python
import netCDF4
import numpy as np
```

9.2 Select year

[2]:
```python
nbyears=10
last_year=np.arange(365*(nbyears-1),365*(nbyears))
last_year.shape
```

[2]: (365,)

[3]:
```python
for i,v in enumerate(last_year):
    print(i,v)
```

0	3285
1	3286
2	3287
3	3288
4	3289
5	3290
6	3291
7	3292
8	3293
9	3294
10	3295
11	3296
12	3297
13	3298
14	3299
15	3300
16	3301
17	3302
18	3303
19	3304
20	3305
21	3306
22	3307
23	3308
24	3309
25	3310
26	3311
27	3312
28	3313
29	3314
30	3315
31	3316
32	3317
33	3318
34	3319
35	3320
36	3321
37	3322
38	3323
39	3324
40	3325
41	3326
42	3327
43	3328
44	3329
45	3330
46	3331
47	3332

48	3333
49	3334
50	3335
51	3336
52	3337
53	3338
54	3339
55	3340
56	3341
57	3342
58	3343
59	3344
60	3345
61	3346
62	3347
63	3348
64	3349
65	3350
66	3351
67	3352
68	3353
69	3354
70	3355
71	3356
72	3357
73	3358
74	3359
75	3360
76	3361
77	3362
78	3363
79	3364
80	3365
81	3366
82	3367
83	3368
84	3369
85	3370
86	3371
87	3372
88	3373
89	3374
90	3375
91	3376
92	3377
93	3378
94	3379
95	3380

96 3381
97 3382
98 3383
99 3384
100 3385
101 3386
102 3387
103 3388
104 3389
105 3390
106 3391
107 3392
108 3393
109 3394
110 3395
111 3396
112 3397
113 3398
114 3399
115 3400
116 3401
117 3402
118 3403
119 3404
120 3405
121 3406
122 3407
123 3408
124 3409
125 3410
126 3411
127 3412
128 3413
129 3414
130 3415
131 3416
132 3417
133 3418
134 3419
135 3420
136 3421
137 3422
138 3423
139 3424
140 3425
141 3426
142 3427
143 3428

144	3429
145	3430
146	3431
147	3432
148	3433
149	3434
150	3435
151	3436
152	3437
153	3438
154	3439
155	3440
156	3441
157	3442
158	3443
159	3444
160	3445
161	3446
162	3447
163	3448
164	3449
165	3450
166	3451
167	3452
168	3453
169	3454
170	3455
171	3456
172	3457
173	3458
174	3459
175	3460
176	3461
177	3462
178	3463
179	3464
180	3465
181	3466
182	3467
183	3468
184	3469
185	3470
186	3471
187	3472
188	3473
189	3474
190	3475
191	3476

192	3477
193	3478
194	3479
195	3480
196	3481
197	3482
198	3483
199	3484
200	3485
201	3486
202	3487
203	3488
204	3489
205	3490
206	3491
207	3492
208	3493
209	3494
210	3495
211	3496
212	3497
213	3498
214	3499
215	3500
216	3501
217	3502
218	3503
219	3504
220	3505
221	3506
222	3507
223	3508
224	3509
225	3510
226	3511
227	3512
228	3513
229	3514
230	3515
231	3516
232	3517
233	3518
234	3519
235	3520
236	3521
237	3522
238	3523
239	3524

240	3525
241	3526
242	3527
243	3528
244	3529
245	3530
246	3531
247	3532
248	3533
249	3534
250	3535
251	3536
252	3537
253	3538
254	3539
255	3540
256	3541
257	3542
258	3543
259	3544
260	3545
261	3546
262	3547
263	3548
264	3549
265	3550
266	3551
267	3552
268	3553
269	3554
270	3555
271	3556
272	3557
273	3558
274	3559
275	3560
276	3561
277	3562
278	3563
279	3564
280	3565
281	3566
282	3567
283	3568
284	3569
285	3570
286	3571
287	3572

288	3573
289	3574
290	3575
291	3576
292	3577
293	3578
294	3579
295	3580
296	3581
297	3582
298	3583
299	3584
300	3585
301	3586
302	3587
303	3588
304	3589
305	3590
306	3591
307	3592
308	3593
309	3594
310	3595
311	3596
312	3597
313	3598
314	3599
315	3600
316	3601
317	3602
318	3603
319	3604
320	3605
321	3606
322	3607
323	3608
324	3609
325	3610
326	3611
327	3612
328	3613
329	3614
330	3615
331	3616
332	3617
333	3618
334	3619
335	3620

```
336  3621
337  3622
338  3623
339  3624
340  3625
341  3626
342  3627
343  3628
344  3629
345  3630
346  3631
347  3632
348  3633
349  3634
350  3635
351  3636
352  3637
353  3638
354  3639
355  3640
356  3641
357  3642
358  3643
359  3644
360  3645
361  3646
362  3647
363  3648
364  3649
```

9.3 Grid

See the documentation on the grid of the output of the MITgcm model: https://darwin3.readthedocs.io/en/latest/getting_started/getting_started.html#grid.

RC is the r coordinate of cell center (in m).

[4]:
```
gridfile='data/grid.t001.nc'
```

[5]:
```
ncfile=gridfile
variable='RC'
try:
    # open the netCDF file for reading
    fh=netCDF4.Dataset(ncfile,'r')
except:
    raise IOError("File not found: {}.".format(ncfile))
try:
```

```python
    # read the data in variable named v
    RC=fh.variables[variable][:]
except:
    raise IOError("Variable {} not found in {}.".format(variable, ncfile))
fh.close()
```

We switch sign to have positive depths.

[6]:
```python
# swith sign
RC=-RC
RC.shape
```

[6]: (75,)

[7]:
```python
for i,v in enumerate(RC):
    print(i,v)
```

```
0 0.5061625
1 1.5333449999999997
2 2.5948719999999996
3 3.7013264999999995
4 4.866235
5 6.106664499999999
6 7.443784999999999
7 8.903281
8 10.5154235
9 12.314566000000001
10 14.337813500000001
11 16.6226945
12 19.203892000000003
13 22.109410500000003
14 25.356929
15 28.9512475
16 32.883559500000004
17 37.1327755
18 41.6684945
19 46.454799
20 51.453981
21 56.6295605
22 61.9483055
23 67.381287
24 72.9041905
25 78.4971245
26 84.1441395
27 89.832647
28 95.5528315
29 101.2971055
30 107.0596445
```

31 112.836004
32 118.622804
33 124.417478
34 130.2180915
35 136.02317200000002
36 141.8316135
37 147.6425875
38 153.4554685
39 159.269784
40 165.08516799999998
41 170.9013685
42 176.718179
43 182.5354395
44 188.35304349999998
45 194.17090699999997
46 199.98896899999997
47 205.80717599999997
48 211.62548949999996
49 217.44388699999996
50 223.26235349999996
51 229.08086549999996
52 234.89940799999997
53 240.71797349999997
54 246.53655399999997
55 252.35515749999996
56 258.17376899999994
57 263.99238799999995
58 269.8110219999999
59 275.6296559999999
60 281.4482899999999
61 287.26692399999985
62 293.08557349999984
63 298.90422299999983
64 304.7228569999998
65 310.5415064999998
66 316.3601559999998
67 322.17878999999976
68 327.99743949999976
69 333.81608899999975
70 339.63473849999974
71 345.45338799999973
72 351.2720374999997
73 357.0906869999997
74 362.9093209999997

9.4 Primary productivity

[8]:
```
ppfile='data/rates.0000000000.t001.nc'
```

PP is the primary production (in mmol C m^{-3} s^{-1}).

[9]:
```
ncfile=ppfile
variable='PP'
try:
    # open the netCDF file for reading
    fh=netCDF4.Dataset(ncfile,'r')
except:
    raise IOError("File not found: {}.".format(ncfile))
try:
    # read the data in variable named v
    array2d_idepth_iT_ppfull=fh.variables[variable][:]
except:
    raise IOError("Variable {} not found in {}.".format(variable, ncfile))
fh.close()
array2d_idepth_iT_ppfull.shape
```

[9]:
```
(3650, 75, 1, 1)
```

[10]:
```
array2d_idepth_iT_ppfull=array2d_idepth_iT_ppfull.squeeze()
array2d_idepth_iT_ppfull.shape
```

[10]:
```
(3650, 75)
```

[11]:
```
array2d_idepth_iT_ppfull=array2d_idepth_iT_ppfull.transpose()
array2d_idepth_iT_ppfull.shape
```

[11]:
```
(75, 3650)
```

[12]:
```
array2d_idepth_iT_pp=array2d_idepth_iT_ppfull[:,last_year]
array2d_idepth_iT_pp.shape
```

[12]:
```
(75, 365)
```

The row is the index of the depth (0-based). The column is the index of the day of year of the tenth year (0-based).

10 Export a NetCDF file into a CSV file with Python

Suppose we want the comma-separated values (CSV) file PP.csv in a wide form (not data matrix form, a.k.a. tidy form) with 366 columns. The first column is the depth in m. The 365 following columns are the days of year. The first row is the headers. The 75 following rows are the depths.

```
depth_m,doy_001,doy_002,...,doy_365
0.5061625,...
...
362.9093209999997,...
```

First, load libraries.

[13]:
```python
import pandas as pd
```

[14]:
```python
df_depth=pd.DataFrame({'depth_m':RC})
df_depth
```

[14]:
```
       depth_m
0     0.506162
1     1.533345
2     2.594872
3     3.701326
4     4.866235
..         …
70  339.634738
71  345.453388
72  351.272037
73  357.090687
74  362.909321

[75 rows x 1 columns]
```

[15]:
```python
df_PP_names=['doy_{0:03}'.format(i) for i in range(1,366)]
df_PP_names
```

[15]:
```
['doy_001',
 'doy_002',
 'doy_003',
 'doy_004',
 'doy_005',
 'doy_006',
 'doy_007',
 'doy_008',
 'doy_009',
 'doy_010',
 'doy_011',
 'doy_012',
 'doy_013',
 'doy_014',
 'doy_015',
 'doy_016',
 'doy_017',
 'doy_018',
```

```
'doy_019',
'doy_020',
'doy_021',
'doy_022',
'doy_023',
'doy_024',
'doy_025',
'doy_026',
'doy_027',
'doy_028',
'doy_029',
'doy_030',
'doy_031',
'doy_032',
'doy_033',
'doy_034',
'doy_035',
'doy_036',
'doy_037',
'doy_038',
'doy_039',
'doy_040',
'doy_041',
'doy_042',
'doy_043',
'doy_044',
'doy_045',
'doy_046',
'doy_047',
'doy_048',
'doy_049',
'doy_050',
'doy_051',
'doy_052',
'doy_053',
'doy_054',
'doy_055',
'doy_056',
'doy_057',
'doy_058',
'doy_059',
'doy_060',
'doy_061',
'doy_062',
'doy_063',
'doy_064',
'doy_065',
```

```
'doy_066',
'doy_067',
'doy_068',
'doy_069',
'doy_070',
'doy_071',
'doy_072',
'doy_073',
'doy_074',
'doy_075',
'doy_076',
'doy_077',
'doy_078',
'doy_079',
'doy_080',
'doy_081',
'doy_082',
'doy_083',
'doy_084',
'doy_085',
'doy_086',
'doy_087',
'doy_088',
'doy_089',
'doy_090',
'doy_091',
'doy_092',
'doy_093',
'doy_094',
'doy_095',
'doy_096',
'doy_097',
'doy_098',
'doy_099',
'doy_100',
'doy_101',
'doy_102',
'doy_103',
'doy_104',
'doy_105',
'doy_106',
'doy_107',
'doy_108',
'doy_109',
'doy_110',
'doy_111',
'doy_112',
```

```
'doy_113',
'doy_114',
'doy_115',
'doy_116',
'doy_117',
'doy_118',
'doy_119',
'doy_120',
'doy_121',
'doy_122',
'doy_123',
'doy_124',
'doy_125',
'doy_126',
'doy_127',
'doy_128',
'doy_129',
'doy_130',
'doy_131',
'doy_132',
'doy_133',
'doy_134',
'doy_135',
'doy_136',
'doy_137',
'doy_138',
'doy_139',
'doy_140',
'doy_141',
'doy_142',
'doy_143',
'doy_144',
'doy_145',
'doy_146',
'doy_147',
'doy_148',
'doy_149',
'doy_150',
'doy_151',
'doy_152',
'doy_153',
'doy_154',
'doy_155',
'doy_156',
'doy_157',
'doy_158',
'doy_159',
```

```
'doy_160',
'doy_161',
'doy_162',
'doy_163',
'doy_164',
'doy_165',
'doy_166',
'doy_167',
'doy_168',
'doy_169',
'doy_170',
'doy_171',
'doy_172',
'doy_173',
'doy_174',
'doy_175',
'doy_176',
'doy_177',
'doy_178',
'doy_179',
'doy_180',
'doy_181',
'doy_182',
'doy_183',
'doy_184',
'doy_185',
'doy_186',
'doy_187',
'doy_188',
'doy_189',
'doy_190',
'doy_191',
'doy_192',
'doy_193',
'doy_194',
'doy_195',
'doy_196',
'doy_197',
'doy_198',
'doy_199',
'doy_200',
'doy_201',
'doy_202',
'doy_203',
'doy_204',
'doy_205',
'doy_206',
```

```
'doy_207',
'doy_208',
'doy_209',
'doy_210',
'doy_211',
'doy_212',
'doy_213',
'doy_214',
'doy_215',
'doy_216',
'doy_217',
'doy_218',
'doy_219',
'doy_220',
'doy_221',
'doy_222',
'doy_223',
'doy_224',
'doy_225',
'doy_226',
'doy_227',
'doy_228',
'doy_229',
'doy_230',
'doy_231',
'doy_232',
'doy_233',
'doy_234',
'doy_235',
'doy_236',
'doy_237',
'doy_238',
'doy_239',
'doy_240',
'doy_241',
'doy_242',
'doy_243',
'doy_244',
'doy_245',
'doy_246',
'doy_247',
'doy_248',
'doy_249',
'doy_250',
'doy_251',
'doy_252',
'doy_253',
```

```
'doy_254',
'doy_255',
'doy_256',
'doy_257',
'doy_258',
'doy_259',
'doy_260',
'doy_261',
'doy_262',
'doy_263',
'doy_264',
'doy_265',
'doy_266',
'doy_267',
'doy_268',
'doy_269',
'doy_270',
'doy_271',
'doy_272',
'doy_273',
'doy_274',
'doy_275',
'doy_276',
'doy_277',
'doy_278',
'doy_279',
'doy_280',
'doy_281',
'doy_282',
'doy_283',
'doy_284',
'doy_285',
'doy_286',
'doy_287',
'doy_288',
'doy_289',
'doy_290',
'doy_291',
'doy_292',
'doy_293',
'doy_294',
'doy_295',
'doy_296',
'doy_297',
'doy_298',
'doy_299',
'doy_300',
```

```
'doy_301',
'doy_302',
'doy_303',
'doy_304',
'doy_305',
'doy_306',
'doy_307',
'doy_308',
'doy_309',
'doy_310',
'doy_311',
'doy_312',
'doy_313',
'doy_314',
'doy_315',
'doy_316',
'doy_317',
'doy_318',
'doy_319',
'doy_320',
'doy_321',
'doy_322',
'doy_323',
'doy_324',
'doy_325',
'doy_326',
'doy_327',
'doy_328',
'doy_329',
'doy_330',
'doy_331',
'doy_332',
'doy_333',
'doy_334',
'doy_335',
'doy_336',
'doy_337',
'doy_338',
'doy_339',
'doy_340',
'doy_341',
'doy_342',
'doy_343',
'doy_344',
'doy_345',
'doy_346',
'doy_347',
```

```
 'doy_348',
 'doy_349',
 'doy_350',
 'doy_351',
 'doy_352',
 'doy_353',
 'doy_354',
 'doy_355',
 'doy_356',
 'doy_357',
 'doy_358',
 'doy_359',
 'doy_360',
 'doy_361',
 'doy_362',
 'doy_363',
 'doy_364',
 'doy_365']
```

[16]:
```
df_PP=pd.DataFrame(array2d_idepth_iT_pp)
df_PP.columns=df_PP_names
df_PP
```

[16]:
```
    doy_001  doy_002  doy_003  doy_004  doy_005       doy_006       doy_007  \
0       0.0      0.0      0.0      0.0      0.0  3.003517e-12  9.562626e-12
1       0.0      0.0      0.0      0.0      0.0  2.746950e-12  9.208227e-12
2       0.0      0.0      0.0      0.0      0.0  2.635478e-12  8.835832e-12
3       0.0      0.0      0.0      0.0      0.0  2.376848e-12  8.455584e-12
4       0.0      0.0      0.0      0.0      0.0  2.267716e-12  8.068476e-12
..      …        …        …        …        …             …             …
70      0.0      0.0      0.0      0.0      0.0  0.000000e+00  0.000000e+00
71      0.0      0.0      0.0      0.0      0.0  0.000000e+00  0.000000e+00
72      0.0      0.0      0.0      0.0      0.0  0.000000e+00  0.000000e+00
73      0.0      0.0      0.0      0.0      0.0  0.000000e+00  0.000000e+00
74      0.0      0.0      0.0      0.0      0.0  0.000000e+00  0.000000e+00

         doy_008       doy_009       doy_010  …  doy_356  doy_357  doy_358  \
0   3.982557e-11  6.567130e-11  6.554848e-11  …      0.0      0.0      0.0
1   3.840060e-11  6.336488e-11  6.324807e-11  …      0.0      0.0      0.0
2   3.689539e-11  6.092277e-11  6.081262e-11  …      0.0      0.0      0.0
3   3.535327e-11  5.841642e-11  5.831308e-11  …      0.0      0.0      0.0
4   3.377870e-11  5.585338e-11  5.575687e-11  …      0.0      0.0      0.0
..             …             …             …  …      …        …
70  0.000000e+00  0.000000e+00  0.000000e+00  …      0.0      0.0      0.0
71  0.000000e+00  0.000000e+00  0.000000e+00  …      0.0      0.0      0.0
72  0.000000e+00  0.000000e+00  0.000000e+00  …      0.0      0.0      0.0
73  0.000000e+00  0.000000e+00  0.000000e+00  …      0.0      0.0      0.0
```

```
74  0.000000e+00  0.000000e+00  0.000000e+00  …       0.0       0.0       0.0

    doy_359  doy_360  doy_361  doy_362  doy_363  doy_364  doy_365
0       0.0      0.0      0.0      0.0      0.0      0.0      0.0
1       0.0      0.0      0.0      0.0      0.0      0.0      0.0
2       0.0      0.0      0.0      0.0      0.0      0.0      0.0
3       0.0      0.0      0.0      0.0      0.0      0.0      0.0
4       0.0      0.0      0.0      0.0      0.0      0.0      0.0
..      …       …       …       …       …       …       …
70      0.0      0.0      0.0      0.0      0.0      0.0      0.0
71      0.0      0.0      0.0      0.0      0.0      0.0      0.0
72      0.0      0.0      0.0      0.0      0.0      0.0      0.0
73      0.0      0.0      0.0      0.0      0.0      0.0      0.0
74      0.0      0.0      0.0      0.0      0.0      0.0      0.0

[75 rows x 365 columns]
```

[17]:

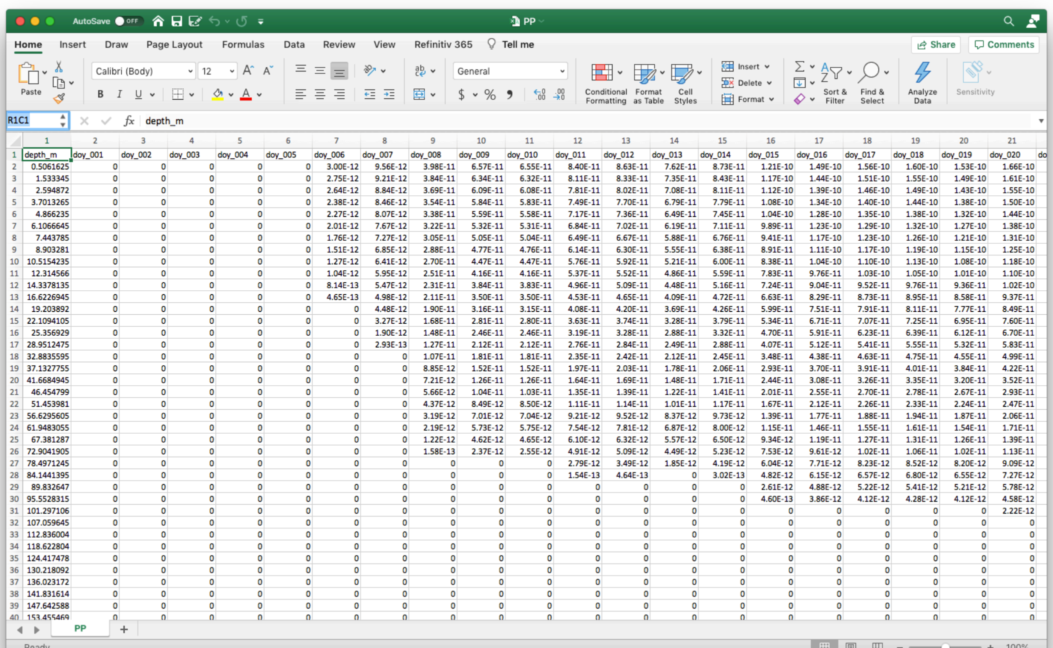
```python
df=pd.concat([df_depth,df_PP],axis='columns')
df
```

[17]:

```
       depth_m  doy_001  doy_002  doy_003  doy_004  doy_005       doy_006  \
0     0.506162      0.0      0.0      0.0      0.0      0.0  3.003517e-12
1     1.533345      0.0      0.0      0.0      0.0      0.0  2.746950e-12
2     2.594872      0.0      0.0      0.0      0.0      0.0  2.635478e-12
3     3.701326      0.0      0.0      0.0      0.0      0.0  2.376848e-12
4     4.866235      0.0      0.0      0.0      0.0      0.0  2.267716e-12
..         …       …       …       …       …       …       …
70  339.634738      0.0      0.0      0.0      0.0      0.0  0.000000e+00
71  345.453388      0.0      0.0      0.0      0.0      0.0  0.000000e+00
72  351.272037      0.0      0.0      0.0      0.0      0.0  0.000000e+00
73  357.090687      0.0      0.0      0.0      0.0      0.0  0.000000e+00
74  362.909321      0.0      0.0      0.0      0.0      0.0  0.000000e+00

         doy_007       doy_008       doy_009  …  doy_356  doy_357  doy_358  \
0   9.562626e-12  3.982557e-11  6.567130e-11  …      0.0      0.0      0.0
1   9.208227e-12  3.840060e-11  6.336488e-11  …      0.0      0.0      0.0
2   8.835832e-12  3.689539e-11  6.092277e-11  …      0.0      0.0      0.0
3   8.455584e-12  3.535327e-11  5.841642e-11  …      0.0      0.0      0.0
4   8.068476e-12  3.377870e-11  5.585338e-11  …      0.0      0.0      0.0
..          …           …           …  …   …       …       …
70  0.000000e+00  0.000000e+00  0.000000e+00  …      0.0      0.0      0.0
71  0.000000e+00  0.000000e+00  0.000000e+00  …      0.0      0.0      0.0
72  0.000000e+00  0.000000e+00  0.000000e+00  …      0.0      0.0      0.0
73  0.000000e+00  0.000000e+00  0.000000e+00  …      0.0      0.0      0.0
74  0.000000e+00  0.000000e+00  0.000000e+00  …      0.0      0.0      0.0

    doy_359  doy_360  doy_361  doy_362  doy_363  doy_364  doy_365
```

```
0     0.0   0.0   0.0   0.0   0.0   0.0   0.0
1     0.0   0.0   0.0   0.0   0.0   0.0   0.0
2     0.0   0.0   0.0   0.0   0.0   0.0   0.0
3     0.0   0.0   0.0   0.0   0.0   0.0   0.0
4     0.0   0.0   0.0   0.0   0.0   0.0   0.0
..    …     …     …     …     …     …     …
70    0.0   0.0   0.0   0.0   0.0   0.0   0.0
71    0.0   0.0   0.0   0.0   0.0   0.0   0.0
72    0.0   0.0   0.0   0.0   0.0   0.0   0.0
73    0.0   0.0   0.0   0.0   0.0   0.0   0.0
74    0.0   0.0   0.0   0.0   0.0   0.0   0.0

[75 rows x 366 columns]
```

[18]:
```python
outfile='PP.csv'
df.to_csv(outfile,index=False)
```



	depth_m	doy_001	doy_002	doy_003	doy_004	doy_005	doy_006	doy_007	doy_008	doy_009
2	0.5061625	0	0	0	0	0	3.00E-12	9.56E-12	3.98E-11	6.57E-11
3	1.533345	0	0	0	0	0	2.75E-12	9.21E-12	3.84E-11	6.34E-11
4	2.594872	0	0	0	0	0	2.64E-12	8.84E-12	3.69E-11	6.09E-11
5	3.7013265	0	0	0	0	0	2.38E-12	8.46E-12	3.54E-11	5.84E-11
6	4.866235	0	0	0	0	0	2.27E-12	8.07E-12	3.38E-11	5.59E-11
7	6.1066645	0	0	0	0	0	2.01E-12	7.67E-12	3.22E-11	5.32E-11
8	7.443785	0	0	0	0	0	1.76E-12	7.27E-12	3.05E-11	5.05E-11
9	8.903281	0	0	0	0	0	1.51E-12	6.85E-12	2.88E-11	4.77E-11
10	10.5154235	0	0	0	0	0	1.27E-12	6.41E-12	2.70E-11	4.47E-11
11	12.314566	0	0	0	0	0	1.04E-12	5.95E-12	2.51E-11	4.16E-11
12	14.3378135	0	0	0	0	0	8.14E-13	5.47E-12	2.31E-11	3.84E-11
13	16.6226945	0	0	0	0	0	4.65E-13	4.98E-12	2.11E-11	3.50E-11
14	19.203892	0	0	0	0	0	0	4.48E-12	1.90E-11	3.16E-11
15	22.1094105	0	0	0	0	0	0	3.27E-12	1.68E-11	2.81E-11
16	25.356929	0	0	0	0	0	0	1.90E-12	1.48E-11	2.46E-11
17	28.9512475	0	0	0	0	0	0	2.93E-13	1.27E-11	2.12E-11
18	32.8835595	0	0	0	0	0	0	0	1.07E-11	1.81E-11
19	37.1327755	0	0	0	0	0	0	0	8.85E-12	1.52E-11
20	41.6684945	0	0	0	0	0	0	0	7.21E-12	1.26E-11
21	46.454799	0	0	0	0	0	0	0	5.66E-12	1.04E-11
22	51.453981	0	0	0	0	0	0	0	4.37E-12	8.49E-12
23	56.6295605	0	0	0	0	0	0	0	3.19E-12	7.01E-12
24	61.9483055	0	0	0	0	0	0	0	2.19E-12	5.73E-12
25	67.381287	0	0	0	0	0	0	0	1.22E-12	4.62E-12
26	72.9041905	0	0	0	0	0	0	0	1.58E-13	2.37E-12
27	78.4971245	0	0	0	0	0	0	0	0	0
28	84.1441395	0	0	0	0	0	0	0	0	0
29	89.832647	0	0	0	0	0	0	0	0	0
30	95.5528315	0	0	0	0	0	0	0	0	0
31	101.297106	0	0	0	0	0	0	0	0	0
32	107.059645	0	0	0	0	0	0	0	0	0
33	112.836004	0	0	0	0	0	0	0	0	0
34	118.622804	0	0	0	0	0	0	0	0	0
35	124.417478	0	0	0	0	0	0	0	0	0
36	130.218092	0	0	0	0	0	0	0	0	0
37	136.023172	0	0	0	0	0	0	0	0	0
38	141.831614	0	0	0	0	0	0	0	0	0
39	147.642588	0	0	0	0	0	0	0	0	0
40	153.455469	0	0	0	0	0	0	0	0	0

	depth_m	doy_010	doy_011	doy_012	doy_013	doy_014	doy_015	doy_016	doy_017	doy_018	doy_019	doy_020
2	0.5061625	6.55E-11	8.40E-11	8.63E-11	7.62E-11	8.73E-11	1.21E-10	1.49E-10	1.56E-10	1.60E-10	1.53E-10	1.66E-10
3	1.533345	6.32E-11	8.11E-11	8.33E-11	7.35E-11	8.43E-11	1.17E-10	1.44E-10	1.51E-10	1.55E-10	1.49E-10	1.61E-10
4	2.594872	6.08E-11	7.81E-11	8.02E-11	7.08E-11	8.11E-11	1.12E-10	1.39E-10	1.46E-10	1.49E-10	1.43E-10	1.55E-10
5	3.7013265	5.83E-11	7.49E-11	7.70E-11	6.79E-11	7.79E-11	1.08E-10	1.34E-10	1.40E-10	1.44E-10	1.38E-10	1.50E-10
6	4.866235	5.58E-11	7.17E-11	7.36E-11	6.49E-11	7.45E-11	1.04E-10	1.28E-10	1.35E-10	1.38E-10	1.32E-10	1.44E-10
7	6.1066645	5.31E-11	6.84E-11	7.02E-11	6.19E-11	7.11E-11	9.89E-11	1.23E-10	1.29E-10	1.32E-10	1.27E-10	1.38E-10
8	7.443785	5.04E-11	6.49E-11	6.67E-11	5.88E-11	6.76E-11	9.41E-11	1.17E-10	1.23E-10	1.26E-10	1.21E-10	1.31E-10
9	8.903281	4.76E-11	6.14E-11	6.30E-11	5.55E-11	6.38E-11	8.91E-11	1.11E-10	1.17E-10	1.19E-10	1.15E-10	1.25E-10
10	10.5154235	4.47E-11	5.76E-11	5.92E-11	5.21E-11	6.00E-11	8.38E-11	1.04E-10	1.10E-10	1.13E-10	1.08E-10	1.18E-10
11	12.314566	4.16E-11	5.37E-11	5.52E-11	4.86E-11	5.59E-11	7.83E-11	9.76E-11	1.03E-10	1.05E-10	1.01E-10	1.10E-10
12	14.3378135	3.83E-11	4.96E-11	5.09E-11	4.48E-11	5.16E-11	7.24E-11	9.04E-11	9.52E-11	9.76E-11	9.36E-11	1.02E-10
13	16.6226945	3.50E-11	4.53E-11	4.65E-11	4.09E-11	4.72E-11	6.63E-11	8.29E-11	8.73E-11	8.95E-11	8.58E-11	9.37E-11
14	19.203892	3.15E-11	4.08E-11	4.20E-11	3.69E-11	4.26E-11	5.99E-11	7.51E-11	7.91E-11	8.11E-11	7.77E-11	8.49E-11
15	22.1094105	2.80E-11	3.63E-11	3.74E-11	3.28E-11	3.79E-11	5.34E-11	6.71E-11	7.07E-11	7.25E-11	6.95E-11	7.60E-11
16	25.356929	2.46E-11	3.19E-11	3.28E-11	2.88E-11	3.32E-11	4.70E-11	5.91E-11	6.23E-11	6.39E-11	6.12E-11	6.70E-11
17	28.9512475	2.12E-11	2.76E-11	2.84E-11	2.49E-11	2.88E-11	4.07E-11	5.12E-11	5.41E-11	5.55E-11	5.32E-11	5.83E-11
18	32.8835595	1.81E-11	2.35E-11	2.42E-11	2.12E-11	2.45E-11	3.48E-11	4.38E-11	4.63E-11	4.75E-11	4.55E-11	4.99E-11
19	37.1327755	1.52E-11	1.97E-11	2.03E-11	1.78E-11	2.06E-11	2.93E-11	3.70E-11	3.91E-11	4.01E-11	3.84E-11	4.22E-11
20	41.6684945	1.26E-11	1.64E-11	1.69E-11	1.48E-11	1.71E-11	2.44E-11	3.08E-11	3.26E-11	3.35E-11	3.20E-11	3.52E-11
21	46.454799	1.03E-11	1.35E-11	1.39E-11	1.22E-11	1.41E-11	2.01E-11	2.55E-11	2.70E-11	2.78E-11	2.67E-11	2.93E-11
22	51.453981	8.50E-12	1.11E-11	1.14E-11	1.01E-11	1.17E-11	1.67E-11	2.12E-11	2.26E-11	2.33E-11	2.24E-11	2.47E-11
23	56.6295605	7.04E-12	9.21E-12	9.52E-12	8.37E-12	9.73E-12	1.39E-11	1.77E-11	1.88E-11	1.94E-11	1.87E-11	2.06E-11
24	61.9483055	5.75E-12	7.54E-12	7.81E-12	6.87E-12	8.00E-12	1.15E-11	1.46E-11	1.55E-11	1.61E-11	1.54E-11	1.71E-11
25	67.381287	4.65E-12	6.10E-12	6.32E-12	5.57E-12	6.50E-12	9.34E-12	1.19E-11	1.27E-11	1.31E-11	1.26E-11	1.39E-11
26	72.9041905	2.55E-12	4.91E-12	5.09E-12	4.49E-12	5.23E-12	7.53E-12	9.61E-12	1.02E-11	1.06E-11	1.02E-11	1.13E-11
27	78.4971245	0	2.79E-12	3.49E-12	1.85E-12	4.19E-12	6.04E-12	7.71E-12	8.23E-12	8.52E-12	8.20E-12	9.09E-12
28	84.1441395	0	1.54E-13	4.64E-13	0	3.02E-13	4.82E-12	6.15E-12	6.57E-12	6.80E-12	6.55E-12	7.27E-12
29	89.832647	0	0	0	0	0	2.61E-12	4.88E-12	5.22E-12	5.41E-12	5.21E-12	5.78E-12
30	95.5528315	0	0	0	0	0	4.60E-13	3.86E-12	4.12E-12	4.28E-12	4.12E-12	4.58E-12
31	101.297106	0	0	0	0	0	0	0	0	0	0	2.22E-12
32	107.059645	0	0	0	0	0	0	0	0	0	0	0
33	112.836004	0	0	0	0	0	0	0	0	0	0	0
34	118.622804	0	0	0	0	0	0	0	0	0	0	0
35	124.417478	0	0	0	0	0	0	0	0	0	0	0
36	130.218092	0	0	0	0	0	0	0	0	0	0	0
37	136.023172	0	0	0	0	0	0	0	0	0	0	0
38	141.831614	0	0	0	0	0	0	0	0	0	0	0
39	147.642588	0	0	0	0	0	0	0	0	0	0	0
40	153.455469	0	0	0	0	0	0	0	0	0	0	0

11 Plot a NetCDF file with Python

First, load libraries.

[19]:
```python
import matplotlib as mpl
import matplotlib.pyplot as plt
```

[20]:
```
plt.close("all")
```

one_year_for_heatmaps is the coordinates on the X-Axis of the corners of quadrilaterals of the pcolormesh (see https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.pcolormesh.html).

[21]:
```
one_year_for_heatmaps=np.arange(0,366)
```

RF_above100 is the coordinates on the Y-Axis of the corners of quadrilaterals of the pcolormesh. We decide to show only the top 100 m.

See the documentation on the grid of the output of the MITgcm model: https://darwin3.readthedocs.io/en/latest/getting_started/getting_started.html#grid.

RF is the r coordinate of cell interface (in m).

[22]:
```
gridfile='data/grid.t001.nc'
```

[23]:
```
ncfile=gridfile
variable='RF'
try:
    # open the netCDF file for reading
    fh=netCDF4.Dataset(ncfile,'r')
except:
    raise IOError("File not found: {}.".format(ncfile))
try:
    # read the data in variable named v
    RF=fh.variables[variable][:]
except:
    raise IOError("Variable {} not found in {}.".format(variable, ncfile))
fh.close()
```

We switch sign to have positive depths.

[24]:
```
# swith sign
RF=-RF
RF.shape
```

[24]: (76,)

[25]:
```
for i,v in enumerate(RF):
    print(i,v)
```

```
0 -0.0
1 1.012325
2 2.0543649999999998
3 3.1353789999999995
4 4.267274
5 5.465196
6 6.748132999999999
```

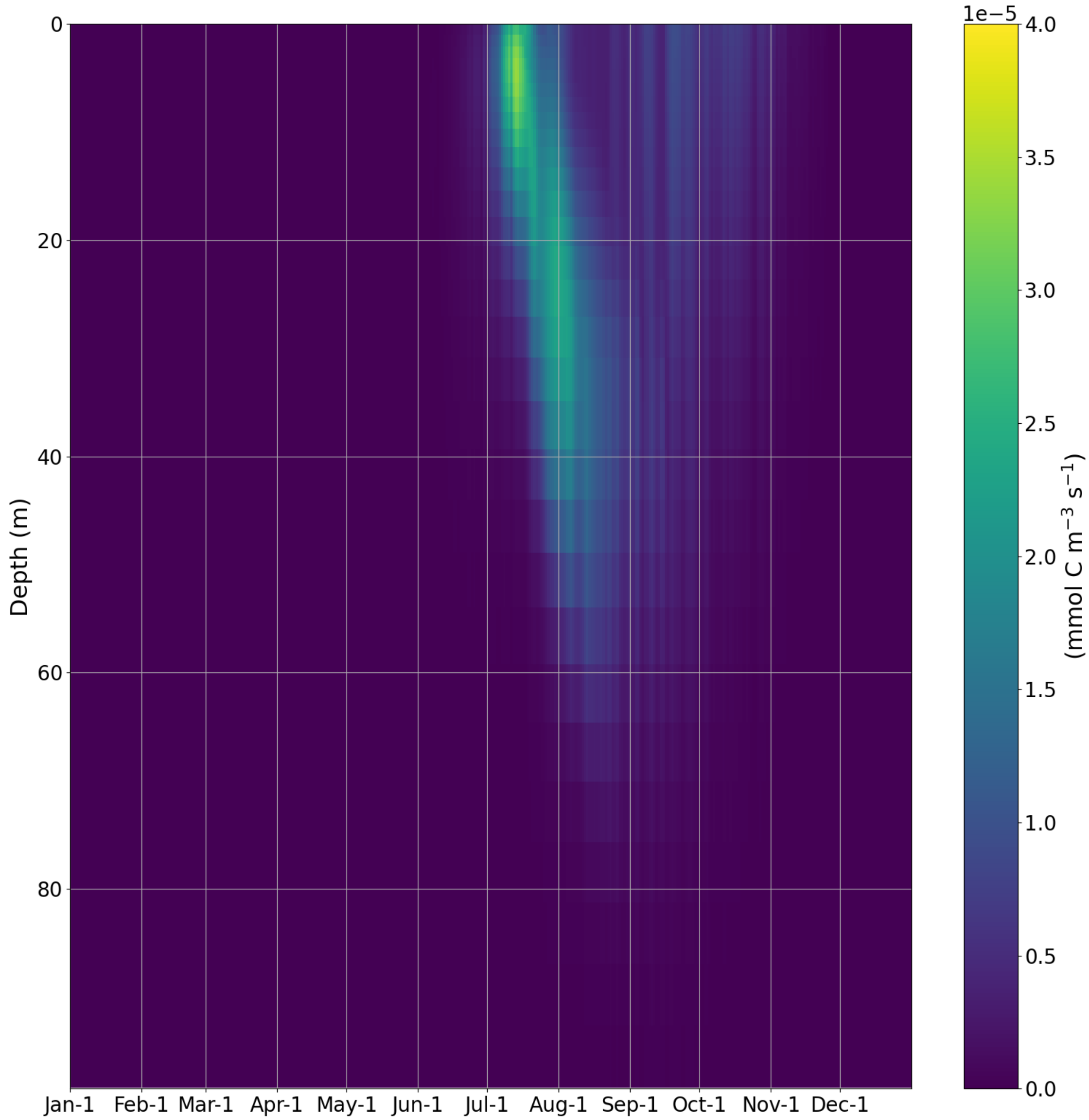
```
7 8.139437
8 9.667124999999999
9 11.363722
10 13.26541
11 15.410217
12 17.835172
13 20.572612
14 23.646209
15 27.067649
16 30.834846
17 34.932272999999995
18 39.33327799999999
19 44.003710999999996
20 48.90588699999999
21 54.00207499999999
22 59.25704599999999
23 64.63956499999999
24 70.123009
25 75.685372
26 81.308877
27 86.979402
28 92.685892
29 98.419771
30 104.17444
31 109.944849
32 115.727159
33 121.518449
34 127.316507
35 133.119676
36 138.926668
37 144.736559
38 150.548616
39 156.362321
40 162.17724700000002
41 167.99308900000003
42 173.80964800000004
43 179.62671000000003
44 185.44416900000004
45 191.26191800000004
46 197.07989600000005
47 202.89804200000006
48 208.71631000000005
49 214.53466900000004
50 220.35310500000003
51 226.17160200000004
52 231.99012900000002
53 237.80868700000002
54 243.62726
```

```
55 249.445848
56 255.26446700000002
57 261.083071
58 266.901705
59 272.72033899999997
60 278.53897299999994
61 284.3576069999999
62 290.1762409999999
63 295.9949059999999
64 301.8135399999999
65 307.63217399999985
66 313.45083899999986
67 319.26947299999983
68 325.0881069999998
69 330.9067719999998
70 336.7254059999998
71 342.5440709999998
72 348.3627049999998
73 354.1813699999998
74 360.00000399999976
75 365.81863799999974
```

[26]:
```
RF_above100=np.array(RF[RF<100])
```

[27]:
```
def make_plots(ax):
    locs=np.array([0, 31, 59, 90, 120, 151,
                   181, 212, 243, 273, 304, 334])
    labels=('Jan-1','Feb-1','Mar-1','Apr-1','May-1','Jun-1',
            'Jul-1','Aug-1','Sep-1','Oct-1','Nov-1','Dec-1')
    h=ax.pcolormesh(one_year_for_heatmaps,
                    RF_above100,
                    array2d_idepth_iT_pp[0:(RF_above100.size)-1,:],
                    cmap='viridis',
                    vmin=0,
                    vmax=0.00004)
    ax.set_xticklabels([])
    ax.set_xticks(locs)
    ax.set_xticklabels(labels)
    ax.set_ylabel('Depth (m)')
    ax.set_ylim(98.5,0)
    ax.grid()
    cbar=plt.colorbar(h)
    cbar.set_label('($\mathrm{ mmol\ C\ m^{-3}\ s^{-1} }$)')

    plt.tight_layout()

with plt.style.context('mplstyles/heatmaps.mplstyle'):
```

```
# Plot
fig=plt.figure(figsize=(16, 16))
ax=fig.add_subplot(111)
make_plots(ax)

# --- SAVE
plt.savefig('figures/PP.png')
plt.show()
```



11.1 More examples in Python

Other examples of Python scripts reading the results of Benoît-Gagné et al. (submitted) are available on https://github.com/maximebenoitgagne/wintertime/blob/v1.5/wintertime.ipynb.