sciris-lightning-slides

July 26, 2024


# Sciris: Simplifying Scientific Python

Cliff Kerr

IDM INSTITUTE FOR DISEASE MODELING

BILL & MELINDA GATES foundation

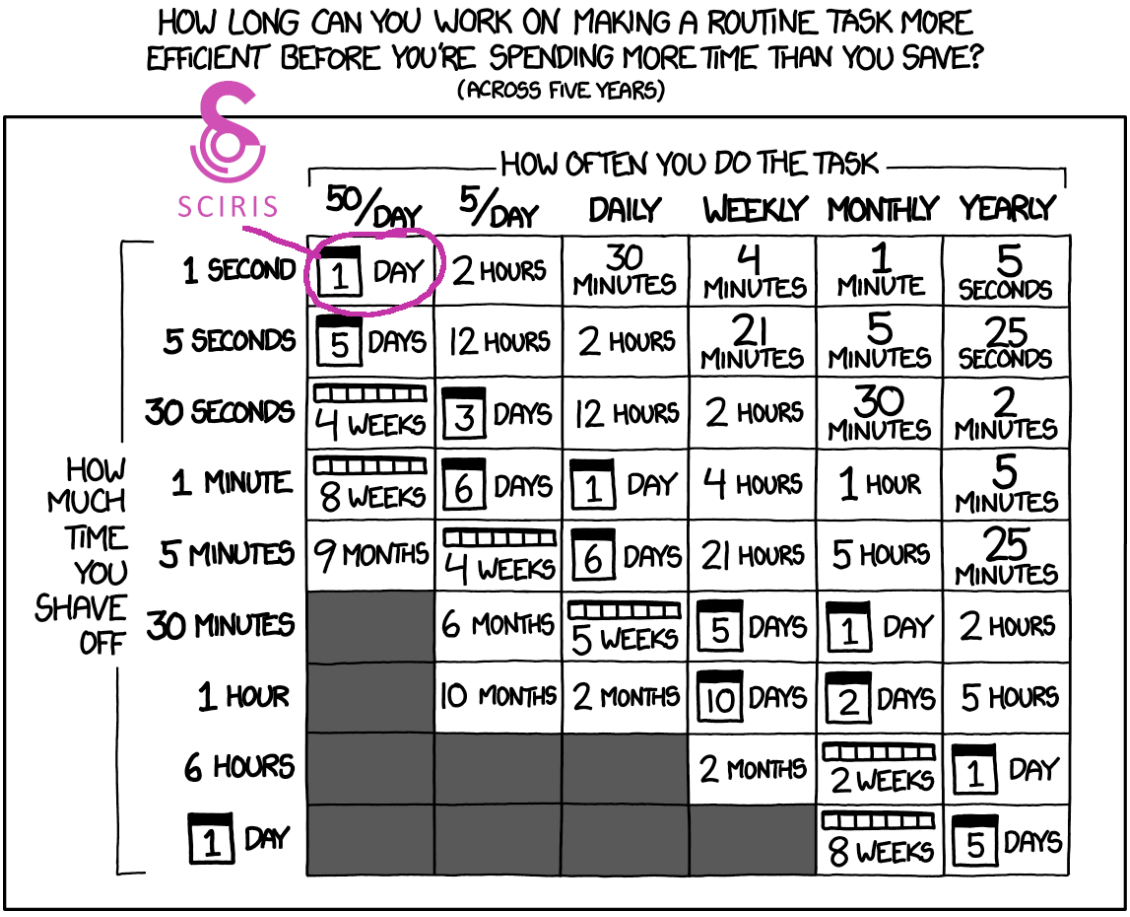


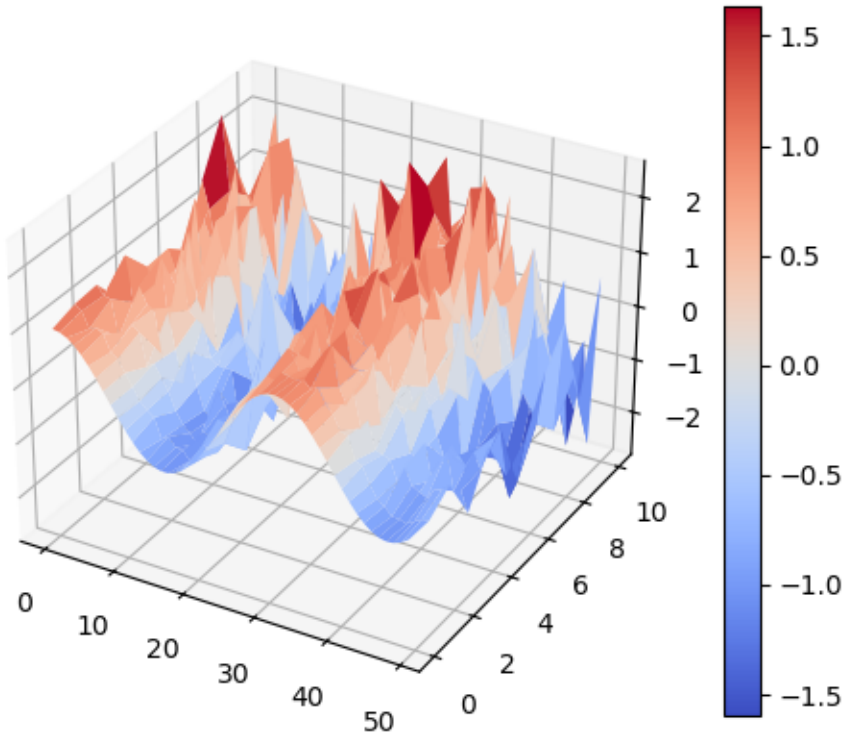

*Image credit: xkcd*

This example illustrates how the same block of fairly typical "scientific Python" code – which performs tasks like collecting data from a function running in parallel, saving and loading files, and 3D plotting – looks like when written in "vanilla Python" compared to using Sciris. See docs.sciris.org for more information.

[2]:
```
#%% Shared code: define the random wave generator

import numpy as np

def randwave(std, xmin=0, xmax=10, npts=50):
    np.random.seed(int(100*std)) # Ensure differences between runs
    a = np.cos(np.linspace(xmin, xmax, npts))
    b = np.random.randn(npts)
    return a + b*std
```

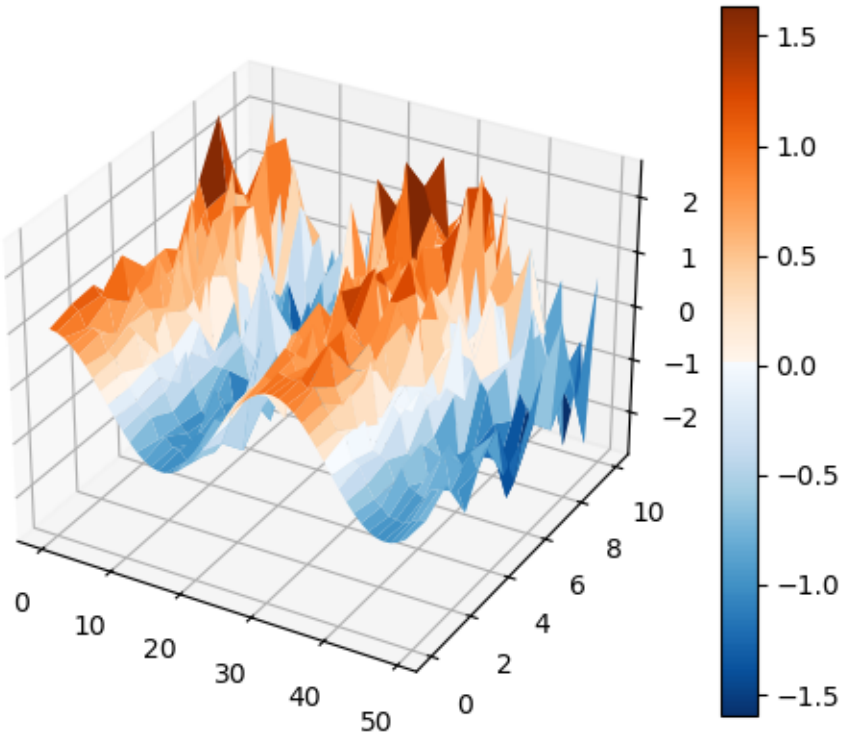
[3]:
```python
#%% Vanilla Python code: 29 lines

# Other imports
import time
import concurrent.futures
import pickle
import gzip
import matplotlib.pyplot as plt

# Start timing
start = time.time()

# Calculate output in parallel
with concurrent.futures.ProcessPoolExecutor() as executor:
    waves = list(executor.map(randwave, np.linspace(0, 1, 11)))

# Save to files
filenames = []
for i,wave in enumerate(waves):
    filename = f'wave{i}.obj'
    with gzip.GzipFile(filename, 'wb') as fileobj:
        fileobj.write(pickle.dumps(wave))
    filenames.append(filename)

# Create dict from files
data_dict = {}
for fname in filenames:
    with gzip.GzipFile(fname) as fileobj:
        filestring = fileobj.read()
        data_dict[fname] = pickle.loads(filestring)
data = np.array([data_dict[fname] for fname in filenames])

# Create 3D plot
fig, ax = plt.subplots(subplot_kw={"projection": "3d"})
ny, nx = np.array(data).shape
x = np.arange(nx)
y = np.arange(ny)
X, Y = np.meshgrid(x, y)
surf = ax.plot_surface(X, Y, data, cmap='coolwarm')
fig.colorbar(surf)

# Print elapsed time
elapsed = time.time() - start
print(f'Elapsed time: {elapsed} s')
```

```
Elapsed time: 0.14980697631835938 s
```




[4]:
```
#%% Sciris equivalent -- 7 lines (4x shorter)

import sciris as sc
T = sc.timer()
waves = sc.parallelize(randwave, np.linspace(0, 1, 11))
filenames = [sc.save(f'wave{i}.obj', wave) for i,wave in enumerate(waves)]
data = sc.odict({fname:sc.load(fname) for fname in filenames})
sc.surf3d(data[:], cmap='orangeblue')
T.toc()
```

```
Elapsed time: 0.143 s
```


0
10
20
30
40
50
0
2
4
6
8
10
2
1
0
−1
−2
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5