Hopfield Error Explanation

Meta

Author: Parmandeep Chaddha.
Date: March 28, 2022.

An Error Example

Consider the following code from the `Hopfield.ipynb` file found here: hopfield.org. Everything upto here seems to be correct.

```
def hopLoop(patt,wts):
    inpatt = patt
    while True:
        rws = list(range(patt.shape[0]))
        cls = list(range(patt.shape[1]))
        r.shuffle(rws)
        r.shuffle(cls)
        testpatt = inpatt
        for rw in rws:
            for cl in cls:
                inpatt[rw][cl] = 1.0 if ((np.reshape(inpatt,(1,inpatt.size))
                                          @ wts[rw]) * inpatt[rw][cl]) > 0 else -1
        if (np.all(testpatt == inpatt)): break
    return(patt)
```

The runner for this code is below. All lines added by me are explained with a comment.

```
import copy # Import the copy module.
r.seed(1234) # Set a seed so that the result is replicable.

myn,mysz = hopGenSzN()
myps = hopMkPatts(myn,mysz)
w = hopMkWts(myps)

new_pats = copy.deepcopy(myps) # Create a deep copy of the patterns.
# Since myps is a `list` or `np.ndarray`s, we CANNOT do `new_pats = myps.copy()`
because this creates a shallow copy. I.e. a new list, but all the arrays inside
the list are references. https://docs.python.org/3/library/copy.html

outps = []
for inp in myps:
    op = hopLoop(inp,w)
    outps.append(op)
```

Now, let's compare the first pattern in the inputs with the first pattern in the outputs. It matches!

```
In [9]: display(myps[0])
        display(outps[0])
        display(myps[0] - outps[0])
```

```
array([[ 1., -1., -1.],
       [ 1.,  1., -1.],
       [-1.,  1.,  1.]])

array([[ 1., -1., -1.],
       [ 1.,  1., -1.],
       [-1.,  1.,  1.]])

array([[0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 0.]])
```

However, this is a bluff. Let's look at the `new_pats[0]`.

```
In [11]: display(new_pats[0])
         display(new_pats[0] - outps[0])
```

```
array([[-1.,  1.,  1.],
       [ 1.,  1., -1.],
       [ 1., -1., -1.]])

array([[-2.,  2.,  2.],
       [ 0.,  0.,  0.],
       [ 2., -2., -2.]])
```

Therefore, the output array is NOT the same as the input array, indicating that there is an error somewhere in the code. **But why does the original function work?** Lets condense the original function down to the main error:

```
def hopLoop(patt,wts):
    inpatt = patt # PROBLEM: inpatt is the same as patt here. It's a name
assignment. Therefore, if we change inpatt, we change PATT!
    while True:
        testpatt = inpatt # Another problem: inpatt=testpatt is a name
assignment to inpatt.
        for rw in rws:
            for cl in cls:
                # modification to inpatt:
                # The problem here is that since testpatt is just a name for
inpatt, testpatt is changed everytime inpatt is changed.
        if (np.all(testpatt == inpatt)): break # Since testpatt is a name for
inpatt, this will always be True.
    return(patt)
```

Remarks

I am stuck. I have no idea how other people completed this assignment...