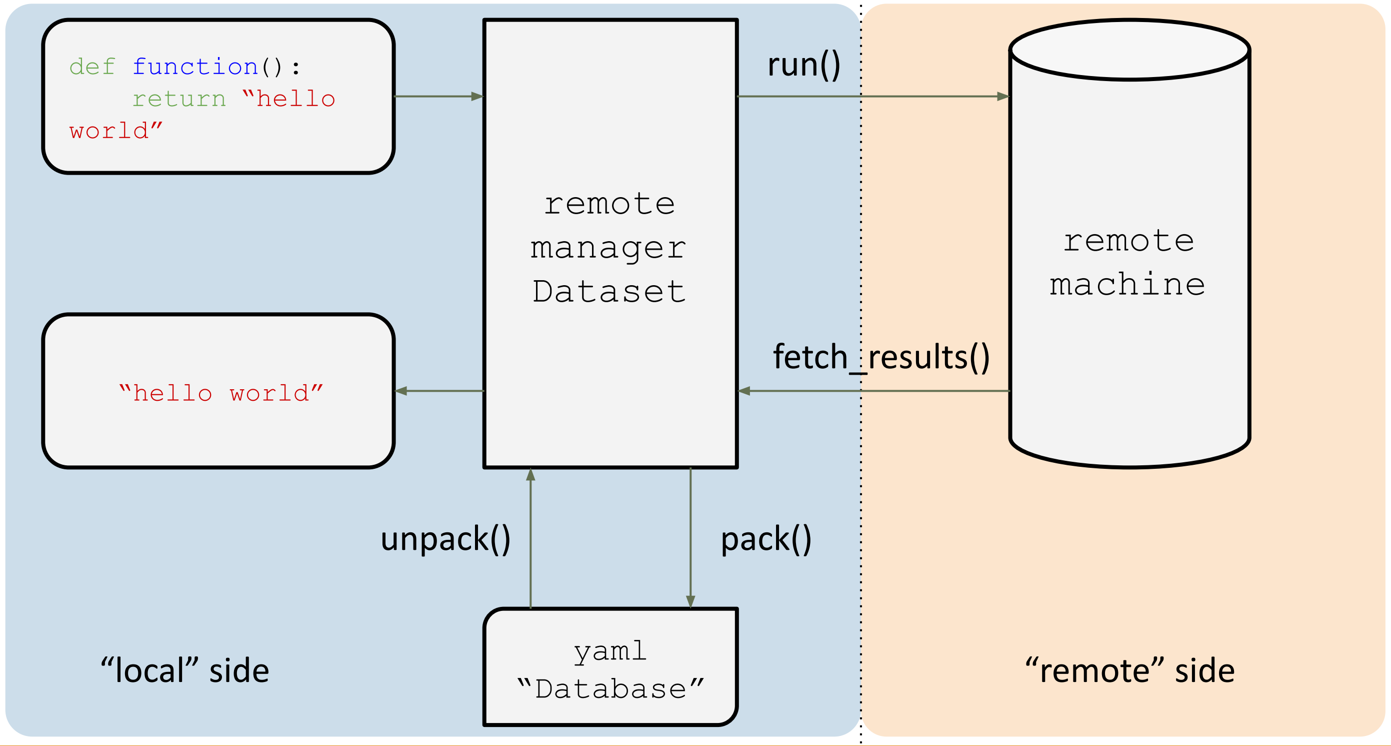
# BigDFT -
# Remote Manager

## Table of Contents

## remotemanager

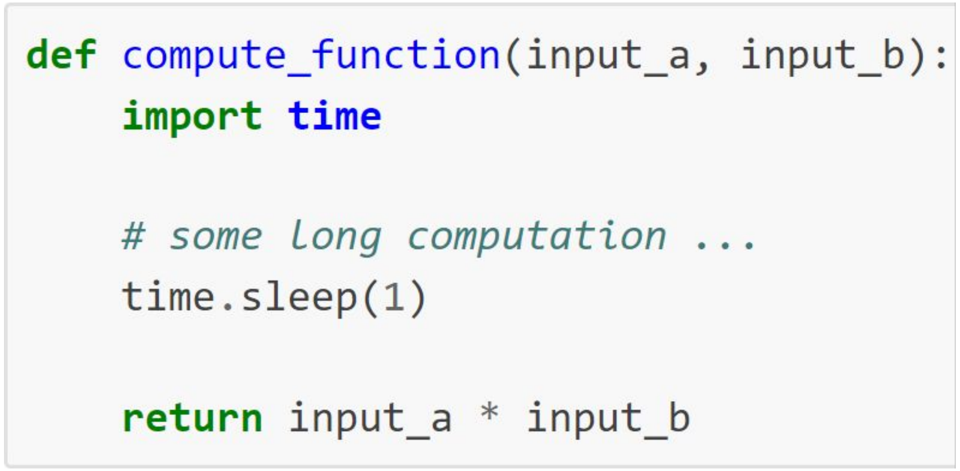
- Workflow tool for interfacing with high performance machines
- Handles job submission and data organisation for you
- Requires a passwordless connection to your machine of choice and a workflow based on python functions

# Intro - remotemanager - overview



## Lightning Tutorial - Function

The function for running is arguably the most important section of your workflow, and the section which will require the most amount of “tinkering”.
It must contain everything you need for your computation, including imports



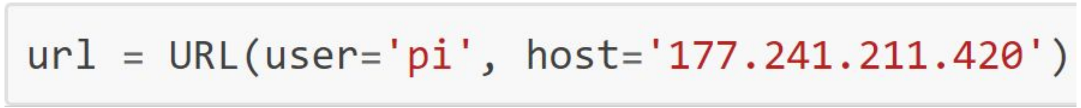
```python
def compute_function(input_a, input_b):
    import time

    # some long computation ...
    time.sleep(1)

    return input_a * input_b
```

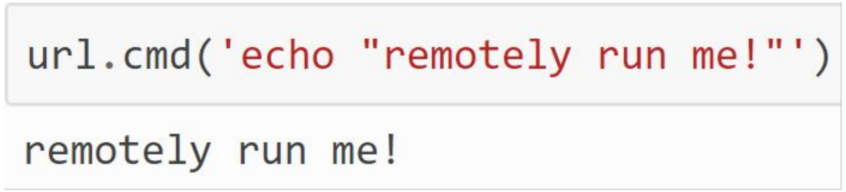
## Lightning Tutorial - Connecting

Next, we must ensure that we can connect to our remote machine



```
url = URL(user='pi', host='177.241.211.420')
```

As an aside, the url object has many functions for housekeeping. A useful one is the cmd method:

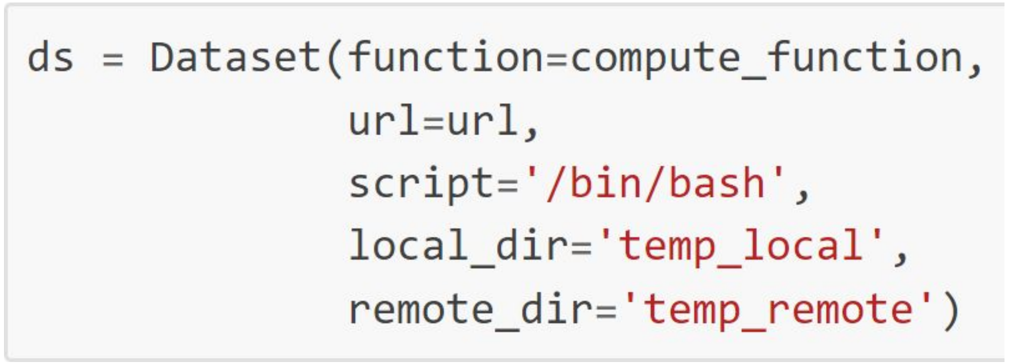


```
url.cmd('echo "remotely run me!"')
```

```
remotely run me!
```

# Lightning Tutorial - Dataset Creation

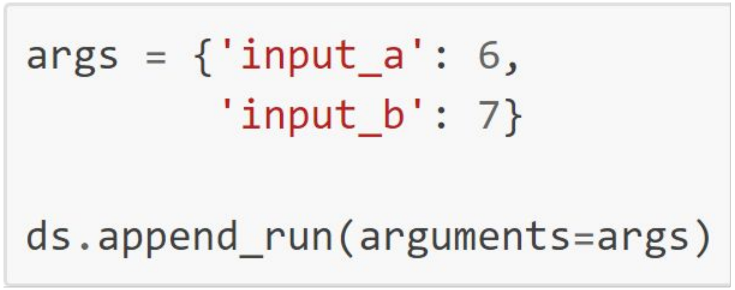
The Dataset is what you will be using to run your functions



```python
ds = Dataset(function=compute_function,
             url=url,
             script='/bin/bash',
             local_dir='temp_local',
             remote_dir='temp_remote')
```

# Lightning Tutorial - Run Creation

The append_run method is used to add a “run” to your function



```python
args = {'input_a': 6,
        'input_b': 7}


ds.append_run(arguments=args)
```

## Lightning Tutorial - Run Creation

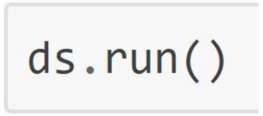
Of course, you are not limited to a single run:

```
runs = [{'input_a': 1, 'input_b': 5},
        {'input_a': 4, 'input_b': 17},
        {'input_a': 9, 'input_b': 14}]

for run in runs:
    ds.append_run(args=run)
```

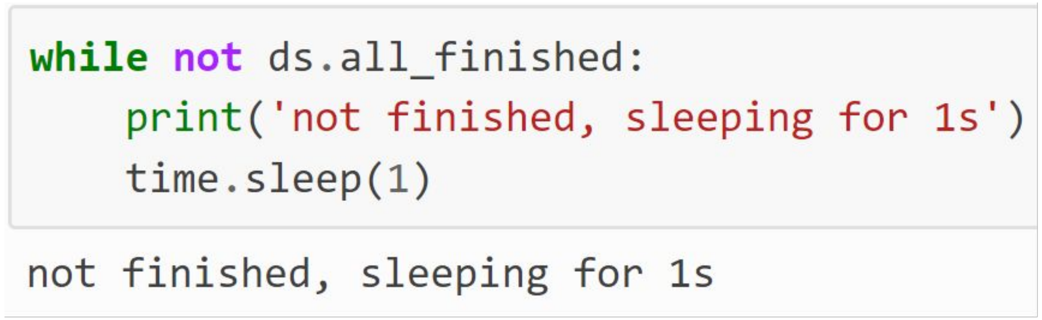
## Lightning Tutorial - Run Execution

Once your jobs are set up, it is simple to run them



```
ds.run()
```

Once we have launched our runs, we now need to wait for them to complete. This notebook cell will block execution until this is the case. You can now shut down your notebook and simply rerun later:

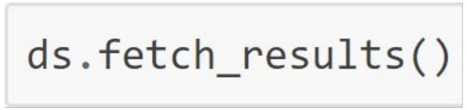


```
while not ds.all_finished:
    print('not finished, sleeping for 1s')
    time.sleep(1)
```

```
not finished, sleeping for 1s
```

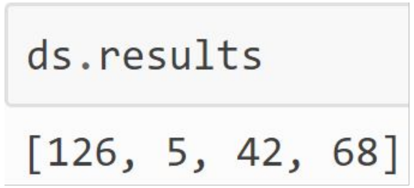
# Lightning Tutorial - Fetching the Results

Once complete, you can fetch your results from the remote machine



```
ds.fetch_results()
```

Which makes them accessible via the results property



```
ds.results
```

```
[126, 5, 42, 68]
```

## Continuation

To follow is a hands on tutorial with the remotemanager systems, this will run through a basic function similar to what we have just seen.

We will run both locally and on Vega