Increasing Awareness of Energy Consumption in Jupyter Notebooks

presented by Marcel

Why?

- Data science and Machine Learning are very power-hungry disciplines
- Jupyter Notebooks are a popular data science tool
- Hides energy consumption
  - Not necessarily local: Server-client architecture
  - Often offered as a service (e.g., Google Colab)

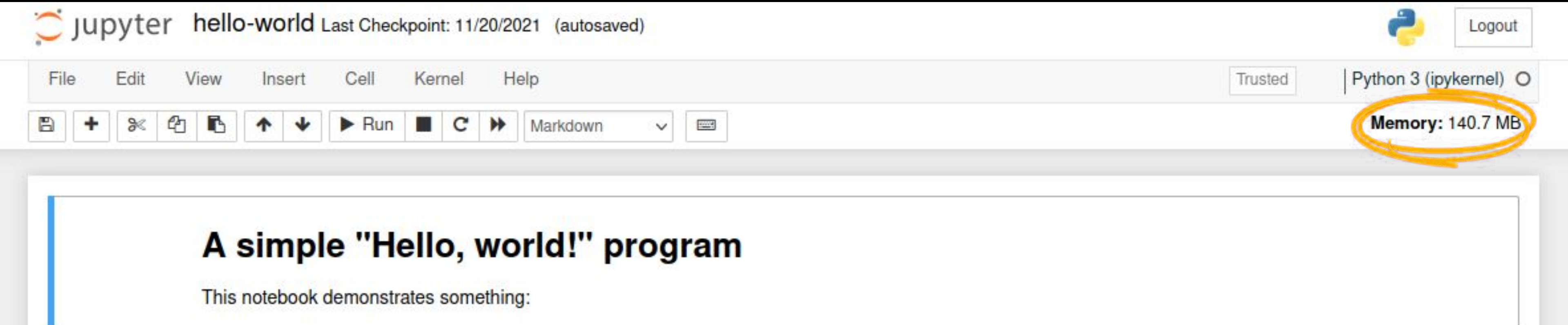
[0]

[0]: en.wikipedia.org/wiki/Project_Jupyter#/media/File:Jupyter_logo.svg

Existing works

- jupyter-resource-usage[0]
- Uses `psutil` to measure RAM and CPU usage
- Other resources not supported (especially not GPU)[1]

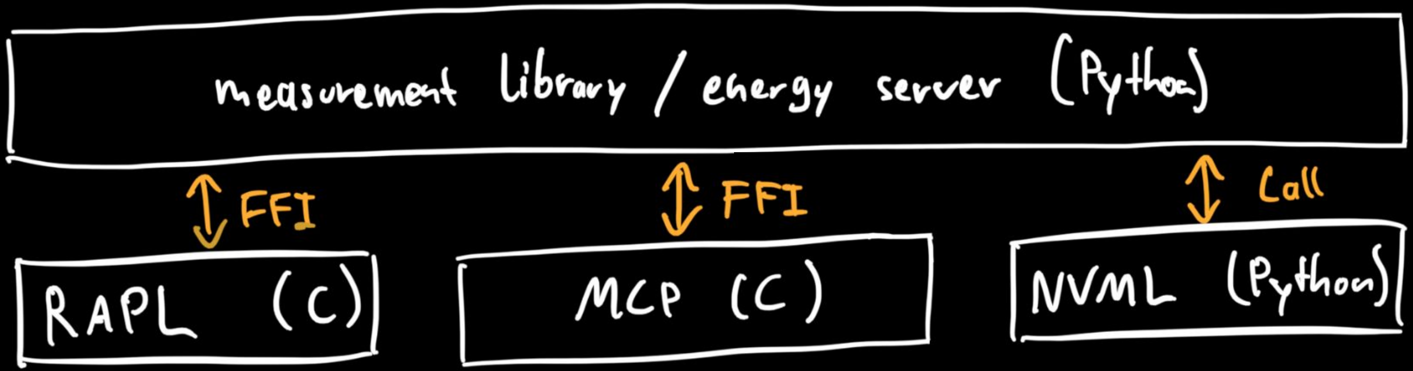
[0]: github.com/jupyter-server/jupyter-resource-usage
[1]: github.com/jupyter-server/jupyter-resource-usage/issues/12



A simple "Hello, world!" program

This notebook demonstrates something:

Sources of energy consumption



“Running Average Power Limit”

CPU, RAM, Internal GPU … of x86_64 CPU components

Covered in my last presentation

“Microchip MCP29F511N Power Monitor”

Hardware intercepting of wall socket and computer

Measures entire system with all components

“Nvidia Management Library”

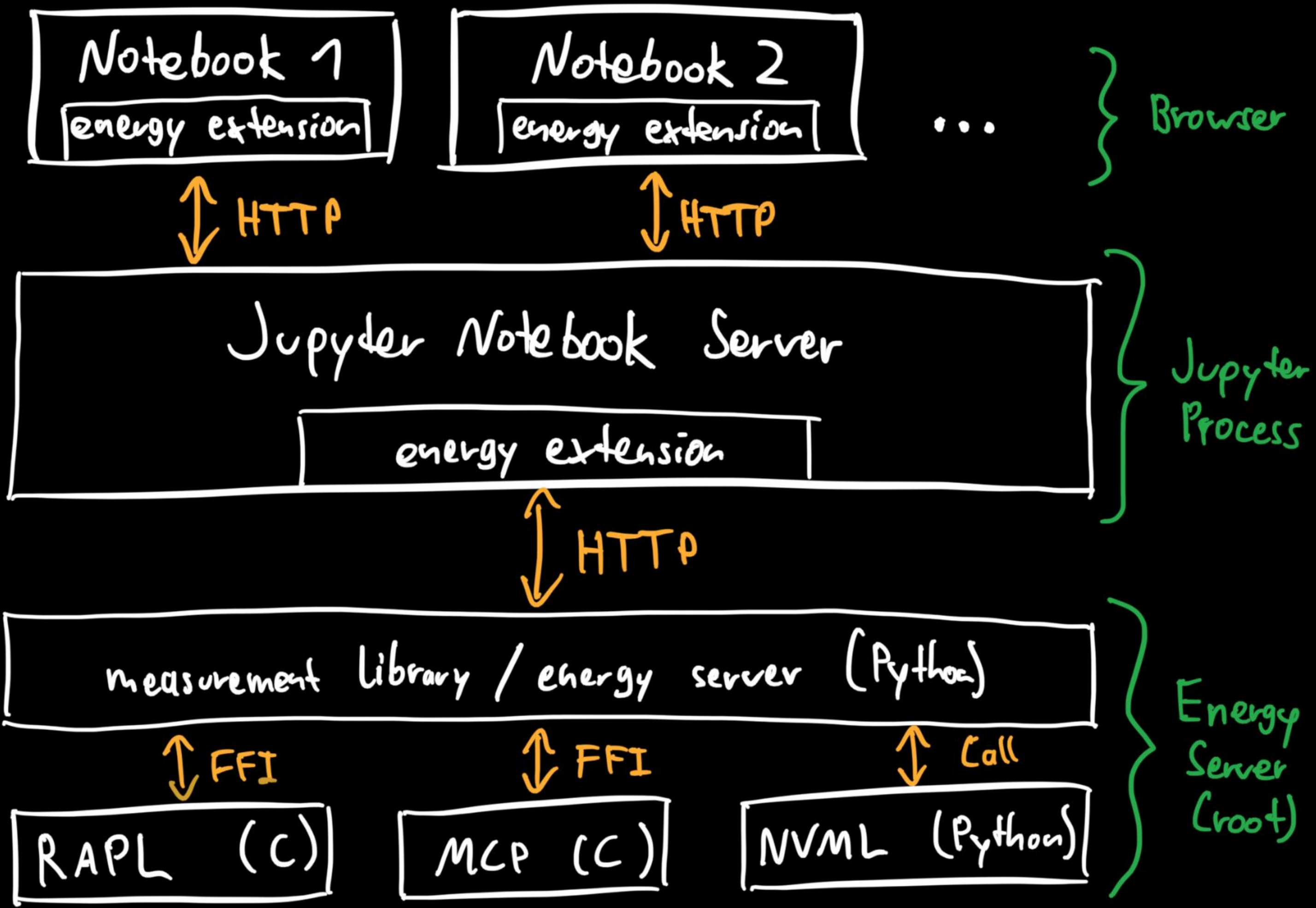
Self-reported energy usage of Nvidia GPUs

`py3nvml` library for Python

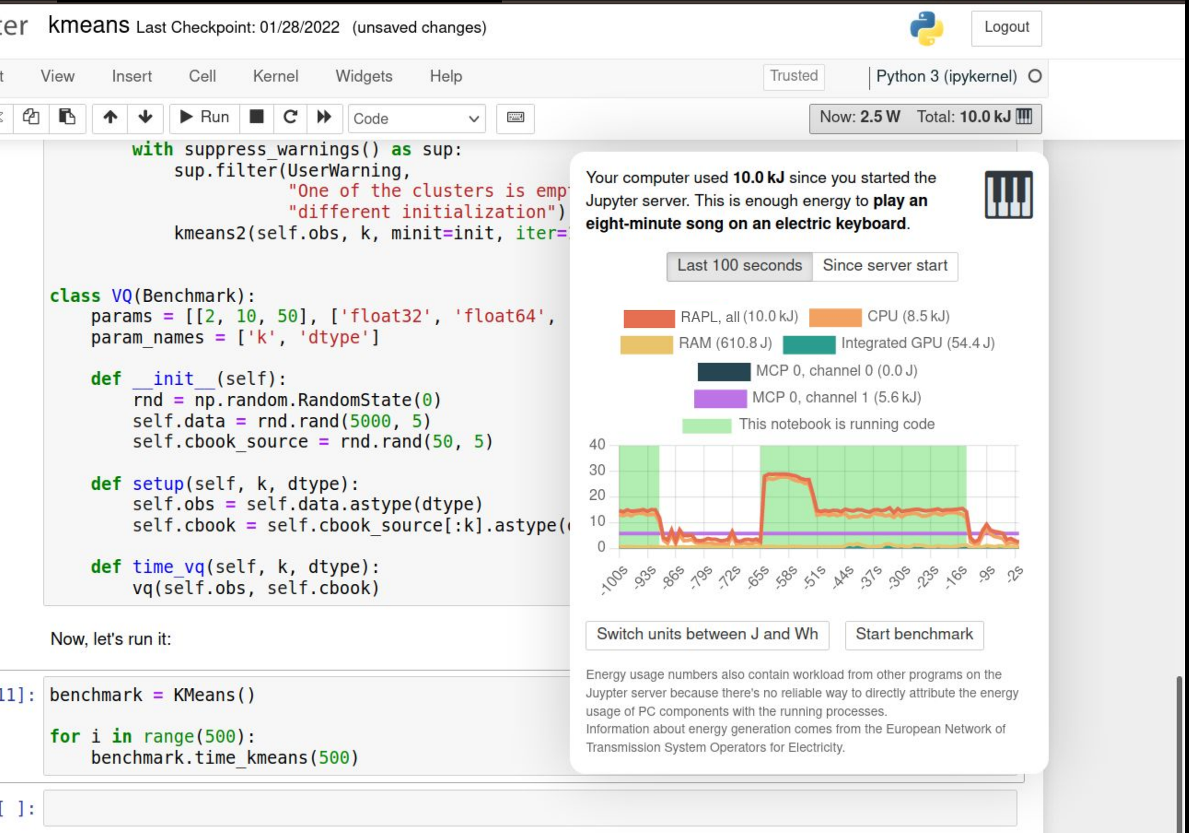
Useful implementation reference: Pinpoint

Can be extended to more sources

Architecture



How it looks



er
kmeans Last Checkpoint: 01/28/2022 (unsaved changes)
Logout
t
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3 (ipykernel)
Run
Code
Now: 2.5 W Total: 10.0 kJ

```
        with suppress_warnings() as sup:
            sup.filter(UserWarning,
                       "One of the clusters is emp
                       "different initialization")
            kmeans2(self.obs, k, minit=init, iter=


class VQ(Benchmark):
    params = [[2, 10, 50], ['float32', 'float64',
    param_names = ['k', 'dtype']

    def __init__(self):
        rnd = np.random.RandomState(0)
        self.data = rnd.rand(5000, 5)
        self.cbook_source = rnd.rand(50, 5)

    def setup(self, k, dtype):
        self.obs = self.data.astype(dtype)
        self.cbook = self.cbook_source[:k].astype(

    def time_vq(self, k, dtype):
        vq(self.obs, self.cbook)
```

Now, let's run it:

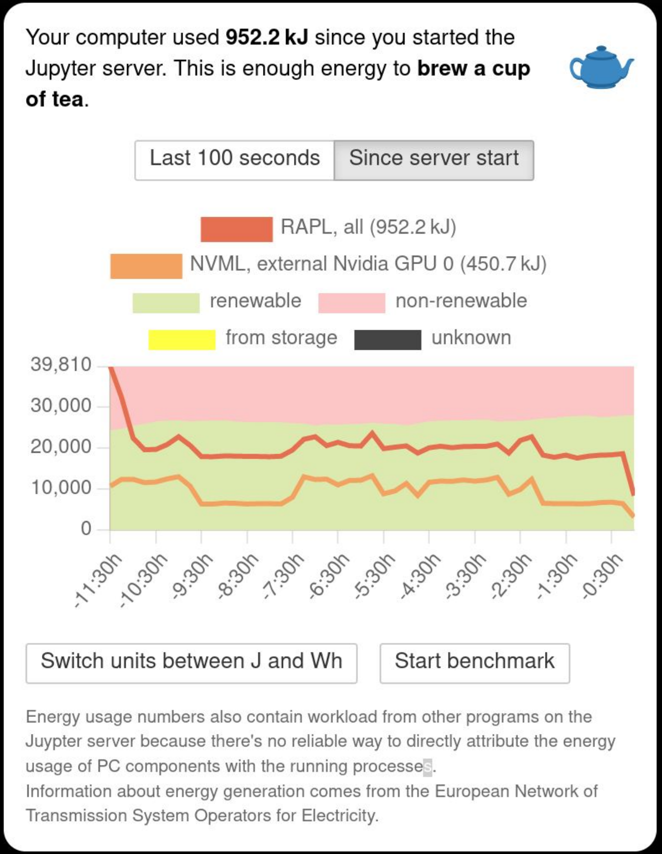
```
11]: benchmark = KMeans()

     for i in range(500):
         benchmark.time_kmeans(500)

[ ]:
```

Your computer used **10.0 kJ** since you started the Jupyter server. This is enough energy to **play an eight-minute song on an electric keyboard**.

Energy usage numbers also contain workload from other programs on the Juypter server because there's no reliable way to directly attribute the energy usage of PC components with the running processes.
Information about energy generation comes from the European Network of Transmission System Operators for Electricity.

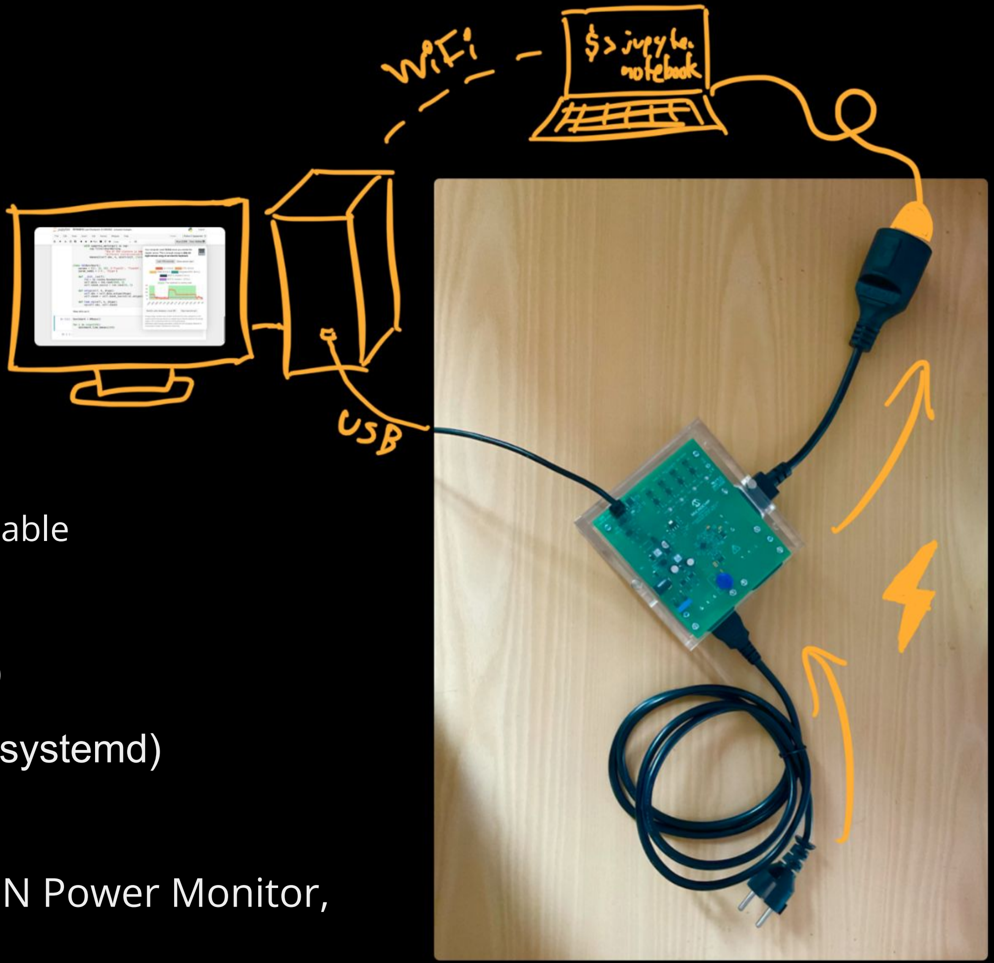


Your computer used **952.2 kJ** since you started the Jupyter server. This is enough energy to **brew a cup of tea**.

Energy usage numbers also contain workload from other programs on the Juypter server because there's no reliable way to directly attribute the energy usage of PC components with the running processes.
Information about energy generation comes from the European Network of Transmission System Operators for Electricity.

Evaluation

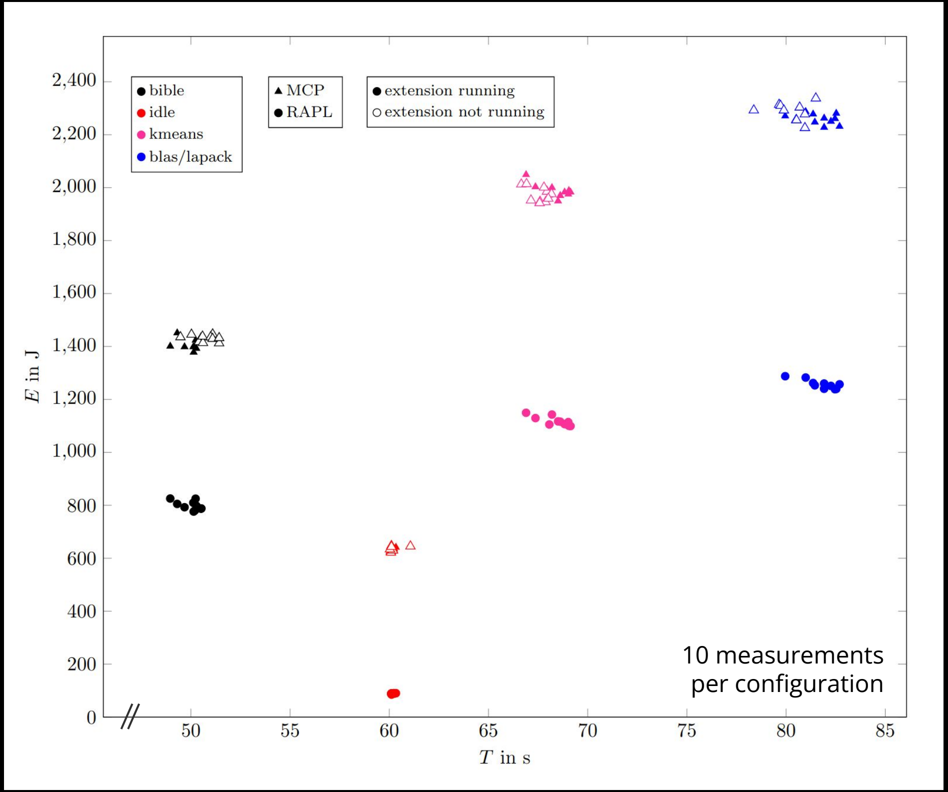
- Asus ZenBook 14, Ubuntu 20.04.3
- Disable some factors that affect energy usage
  - No power savings (BIOS)
  - Fully charged
  - No plugs (USB, HDMI) except power cable
  - Full screen and keyboard brightness
  - Bluetooth off
  - No GUI (everything in terminal mode)
- Existing factors: wifi & critical services (systemd)
- `numactl -C 1 jupyter notebook`
- Measure using Microchip MCP39F511N Power Monitor, USB readout from other computer



Benchmarks

- Idle (Notebook with `sleep(60)`)
  Baseline of energy usage
- K-Means
  Common clustering approach from data science
  memory-bound
  From scipy benchmark
- BLAS/LAPACK (Basic Linear Algebra Subprograms & Linear Algebra PACKage)
  Set of Linear Algebra libraries implemented for many languages
  From scipy benchmark
- Bible
  Querying n-grams search terms against a rudimentary database index
  From Information Retrieval seminar

Results



10 measurements
per configuration

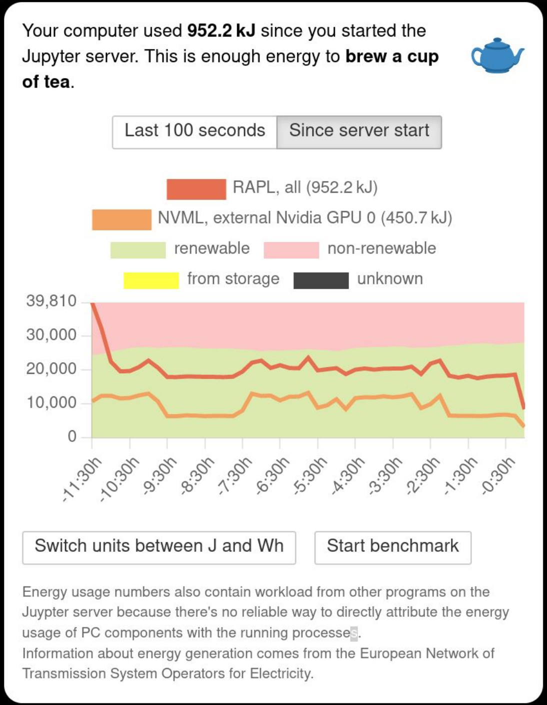
RAPL (internal) vs. MCP (external): energy usage 1.2 to 2.0 times higher (excluding idle)
→ Internal only measures CPU component
→ In idle, CPU is small part of energy usage
(external measurement ~7 times higher)

With vs. without the extension: no significant change in energy use (-0.7 %) or time (+0.6 %, excluding idle, +1.5 % excluding idle & bible)
→ Extension supports everyday usage without huge energy/performance impacts

Noisiness of benchmarks: Idle not noisy. Other benchmarks more noisy with increasing runtime.

Future Work

- How much does the energy cost me on my electricity bill?
- Power generation estimation for other countries & more local
- More variety of generation information: wind, solar, etc. (information already available)
- Make installation easier
- User studies: Does this make people more aware of their energy consumption? Do they actually use less energy?
  Control group: Hidden plugin measures silently, results only disclosed to investigators



Your computer used **952.2 kJ** since you started the Jupyter server. This is enough energy to **brew a cup of tea**.

Energy usage numbers also contain workload from other programs on the Juypter server because there's no reliable way to directly attribute the energy usage of PC components with the running processes.
Information about energy generation comes from the European Network of Transmission System Operators for Electricity.

github.com/MarcelGarus/jupyter-energy
mgar.us/jupyter-energy-slides