LA-UR-24-28950

Approved for public release; distribution is unlimited.

Title:	Leveraging Python for Enhanced MCNP Input and Output Management
Author(s):	Lemke, Zachariah Rudy
Intended for:	2024 MCNP User Symposium, 2024-08-19/2024-08-22 (Los Alamos, New Mexico, United States)
Issued:	2024-08-19

Los Alamos NATIONAL LABORATORY

Leveraging Python for Enhanced MCNP Input and Output Management

Zach Lemke, NEN-2

August 22, 2024

Motivation

- mcnp_pstudy has limitations.
- The use of python's various loops allows cell cards to be constructed dynamically.
- Python can read and extract information from output files for:
  - Easier manual review
  - Automated table construction
  - Recursive input file writing

mcnp_pstudy Overview

- mcnp_pstudy has its place.
- Variables are defined using comments (C @@@ varName = value).
- Variables (which can contain multiple values) replace numbers elsewhere in the input deck.

```
gdv
c
1  100 -18.74  -1   imp:n=1
2  0            1   imp:n=0

1     so 8.741

kcode 10000  1.0  15  115
ksrc  0 0 0
m100  92235 -94.73   92238 -5.27
prdmp 0 0 1 1 0
```

```
gdv-A
C @@@  RADIUS = 8.500  8.741  8.750
1  100 -18.74  -1   imp:n=1
2  0            1   imp:n=0

1     so  RADIUS

kcode 10000  1.0  15  115
ksrc  0 0 0
m100  92235 -94.73   92238 -5.27
prdmp 0 0 1 1 0
```

mcnp_pstudy Other Capabilities

- Random number generation with multiple PDF options
- Significant mathematical function capability
- Some capability for collecting output data (tallies and k_{eff})

So, why bother with python?

- Flexibility
  - When new functionality is added to MCNP, it isn't necessarily compatible with mcnp_pstudy.
    - New location for additional digit on k_{eff}
- Data Organization
  - Dictionaries allow for the organized storage of large quantities of data while remaining readable.
  - Logical filenames instead of "case001" etc.
- Loops!
  - Some "looping capability" exists in the form of parameter expansion.
  - Significantly less readable once python loop syntax is understood.

Python Overview

- Per Wikipedia: “Python is dynamically typed and garbage-collected. It supports multiple programming paradigms ... It is often described as a "batteries included" language due to its comprehensive standard library.”

- What does this mean for us?
  - It is easy to use.
  - Fewer lines of user written code are needed.
  - Many open source packages already exist.

Python Readability

- Calculate the sum of numbers 1 to 5:

C++

```cpp
#include <iostream>

int main() {
    int sum = 0;
    for (int i = 1; i <= 5; ++i) {
        sum += i;
    }
    std::cout << "The sum is: " << sum << std::endl;
    return 0;
}
```

python

```python
sum = 0
for i in range(1, 6):
    sum += i
print("The sum is:", sum)
```

python takes fewer lines and is easier to read!

Brief Introduction to Loops

while loop

```
# Initialize the starting number
number = 1

# Loop while the number is less than or equal to 5
while number <= 5:
    # Print the current number
    print("The current number is:", number)

    # Increase the number by 1
    number += 1
```

for loop

```
# Loop through a sequence of numbers from 1 to 5
for current_number in range(1, 6):
    # Print the current number
    print("The current number is:", current_number)
```

Why do I love loops?

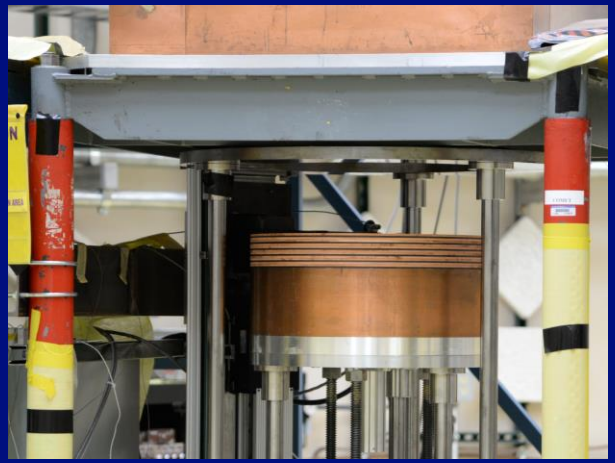
- In short: The ability to write an arbitrary number of cells and associated surface cards depending on my needs.

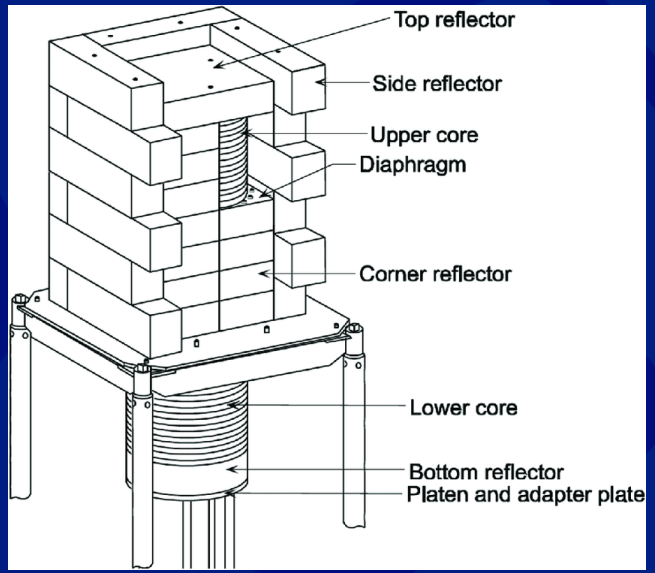
- Flexibility associated with this means new and updated input files can be written at a moment's notice.

- Example: Inputs for the Critical Experiment Reflected By copper to bEtteR Understand Scattering (CERBERUS)

A Quick Detour: CERBERUS

- Zeus style experiment executed at NCERC in Summer 2023
- Performed on the Comet vertical lift assembly
- Targeted Cu elastic and inelastic scattering





An Example from Recent Work

- Dimensional perturbations were performed on all components.
- Density was held constant to keep component masses constant.
  - Component masses were perturbed separately
- Resulted in approximately 700 inputs just for the core.
- Loops allowed the inputs to be generated quickly and without fear of arithmetic error.

Python Dictionaries

- Efficient and readable data storage type.
- Utilizes key-value pairs to store data.
  - This is a nested dictionary
- Easy to reference later in the script.
  - Single dictionary: dictionary[item]
  - Nested dictionary: dictionary[item][subitem]
  - Example: HEUdims[“HEU_01”][“Mass”]

```
HEUdims = {
  "HEU_01" : {
    "OD" : 21.0, # (in)
    "ID" : 15.0, # (in)
    "Height" : 0.30233, # (cm)
    "matCard" : "601",
    "Mass" : 6110.7, # (g)
    "atomDensity" : 4.7773e-2 #atoms per b-cm
  },
  "HEU_02" : {
    "OD" : 21.0, # (in)
    "ID" : 15.0, # (in)
    "Height" : 0.30033, # (cm)
    "matCard" : "602",
    "Mass" : 6109.0, # (g)
    "atomDensity" : 4.8079e-2 #atoms per b-cm
  },
```

Running Input Files from a Python Script

- The os library allows us to interact with a console.

```
os.system(f"mcnp6 i={infilename} o={outfilename}")
```

- Treat what is inside the parentheses exactly like how you would write into the normal MCNP6 terminal.
- All normal options one would expect to find in the input line can be utilized here.

Retrieving Data from an Output File

- The output file can be opened with the open() function.
- Need to identify a string that is unique to shortly before the data of interest.
- rstrip() is used to retrieve text from the file.

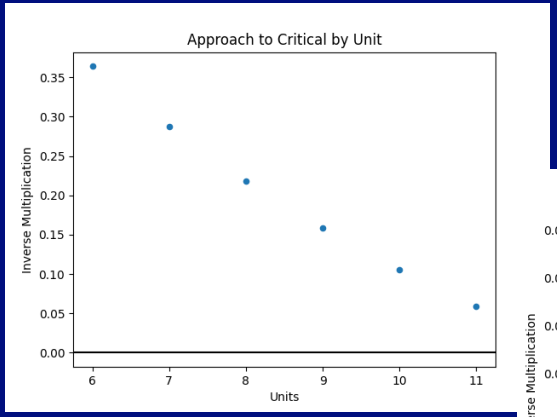
Example: Retrieve k_{eff}

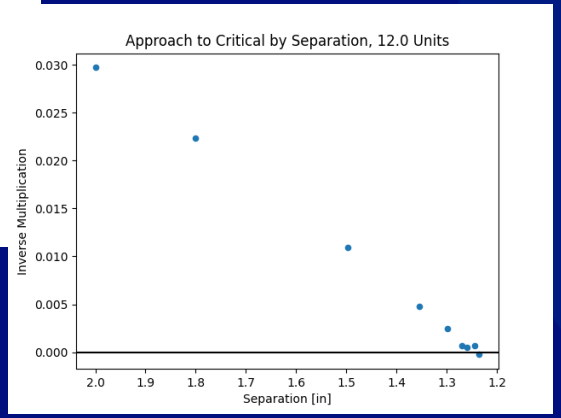
```
with open(outfilename) as output:
    for line in output:
        searchCriteria = 'final result'
        if searchCriteria in line:
            keff = float(line.rstrip()[27:34])
            stddev = float(line.rstrip()[44:51])
```

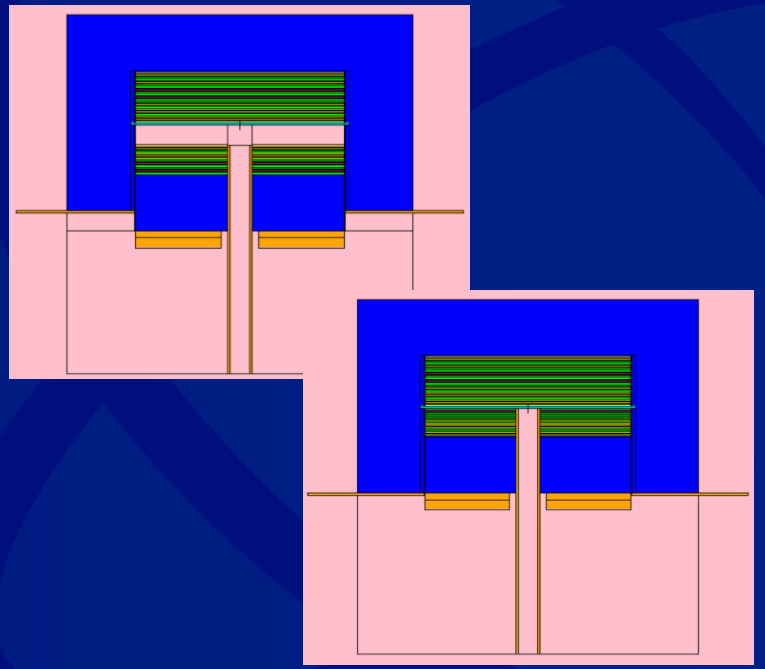
```
searchCriteria = "final estimated"
found = False
with open(filename) as output:
    for line in output:
        if searchCriteria in line:
            keff = float(line.rstrip()[73:81])
            stddev = float(line.rstrip()[122:130])
            found = True
```

Use of Recursive Input Writing

- Input generates output which can then be used to inform the creation of the next input.
- Example: Approach to Critical for CERBERUS







Los Alamos NATIONAL LABORATORY

That’s Just the Beginning

- This barely scratches the surface of what is possible
- Combining modern programming with MCNP allows for significant increases in efficiency, particularly in repetitive tasks
- For example, running approaches to critical for Zeus-like assemblies for all solid elements

Acknowledgments

This work was supported by the US Department of Energy through the Los Alamos National Laboratory. Los Alamos National Laboratory is operated by Triad National Security, LLC, for the National Nuclear Security Administration of the US Department of Energy under Contract No. 89233218CNA000001.

Thank You!