Hackathon

Dr. Heather L. Kline
National Institute of Aerospace
August 9th, 2019

Hackathon: create a python script that….

- Generates an aerodyanmics database:
  - Given a table of flight conditions (Mach, AoA, pressure, temperature)
  - Outputs a table of results including: Cl, Cd, Cmz
  - Optionally also output solution time and Res_Flow[0]
  - Expected syntax:
    hackathon.py -f <input.cfg> -t <trajectory.csv> -i <niter> -o <output.csv>
  - Output file must be in the same format as output_template.csv
  - Rename your script to your_name.py
  - Must keep and use input options for filename and iterations
  - Must accept an arbitrary .csv trajectory file and SU2 configuration file.
- Source files:
  - hackathon_template.py
  - Trajectory.csv
  - output_template.csv

Template File Anatomy

```
# imports
import numpy as np
import csv
import matplotlib.pyplot as plt
from optparse import OptionParser
import os, sys, shutil, copy, os.path
sys.path.append(os.environ['SU2_RUN'])
import SU2

def main():
# Command Line Options
    parser = OptionParser()
    parser.add_option("-f", "--file", dest="filename",
                      help="read config from FILE", metavar="FILE")
    parser.add_option("-n", "--partitions", dest="partitions", default=2,
                      help="number of PARTITIONS", metavar="PARTITIONS")
    parser.add_option("-i", "--iterations", dest="iterations", default=99999,
                      help="number of ITERATIONS", metavar="ITERATIONS")


    (options, args)=parser.parse_args()
    options.partitions = int( options.partitions )
    options.iterations = int( options.iterations )

    # load config, start state
    config = SU2.io.Config(options.filename)
    config.NUMBER_PART = options.partitions
    config.NZONES      = 1
    state  = SU2.io.State()

    # find solution files if they exist
    state.find_files(config)

    # prepare config
    config.NUMBER_PART = options.partitions
    config.EXT_ITER    = options.iterations

    # Initialize results arrays
    nMach = 5
    MachList=np.linspace(0.5,0.6,nMach)
    LiftList=[]
    DragList=[]

    # Output file
    f = open('output.csv', 'w')
    f.write(' Mach, CL, CD \n')

    #with open(options.traject) as csvfile:
    #    myreader = csv.reader(csvfile, delimiter=',')
    #    next(myreader, None)
    #    for row in myreader:
    #        Mach.append(row[1])
    #        Aoa.append(row[0])

    # iterate on Mach number
    for i in range(len(MachList)):
        # local config and state
        konfig = copy.deepcopy(config)
        konfig.DISCARD_INFILES = 'YES'
        ztate  = copy.deepcopy(state)

        # set config options
        konfig.MACH_NUMBER = MachList[i]
        caseName='DIRECT_i_'+str(i)

        # run su2
        drag = SU2.eval.func('DRAG',konfig,ztate)
        lift = SU2.eval.func('LIFT',konfig,ztate)

        LiftList.append(lift)
        DragList.append(drag)

        # format output and write to file
        output = str(MachList[i])+", "\
        + str(lift) + ", " + str(drag) + "\n"
        f.write(output)

        # Store result in a subdirectory
        if os.path.isdir(caseName):
            os.system('rm -rf '+caseName)
        os.system('mv DIRECT '+caseName)

    # Close output file
    f.close()

    # plotting
    plt.figure()
    plt.plot( DragList,LiftList)
    plt.xlabel('Drag Coeffient')
    plt.ylabel('Lift Coefficient')
    plt.savefig('plot.png')
    #plt.show()
```

Define and parse command line options (-f filename, etc)

```
# imports
import numpy as np
import csv
import matplotlib.pyplot as plt
from optparse import OptionParser
import os, sys, shutil, copy, os.path
sys.path.append(os.environ['SU2_RUN'])
import SU2

def main():
# Command Line Options
    parser = OptionParser()
    parser.add_option("-f", "--file", dest="filename",
                      help="read config from FILE", metavar="FILE")
    parser.add_option("-n", "--partitions", dest="partitions", default=2,
                      help="number of PARTITIONS", metavar="PARTITIONS")
    parser.add_option("-i", "--iterations", dest="iterations", default=99999,
                      help="number of ITERATIONS", metavar="ITERATIONS")


    (options, args)=parser.parse_args()
    options.partitions = int( options.partitions )
    options.iterations = int( options.iterations )

    # load config, start state
    config = SU2.io.Config(options.filename)
    config.NUMBER_PART = options.partitions
    config.NZONES      = 1
    state  = SU2.io.State()

    # find solution files if they exist
    state.find_files(config)

    # prepare config
    config.NUMBER_PART = options.partitions
    config.EXT_ITER    = options.iterations

    # Initialize results arrays
    nMach = 5
    MachList=np.linspace(0.5,0.6,nMach)
    LiftList=[]
    DragList=[]

    # Output file
    f = open('output.csv', 'w')
    f.write(' Mach, CL, CD \n')

    #with open(options.traject) as csvfile:
    #    myreader = csv.reader(csvfile, delimiter=',')
    #    next(myreader, None)
    #    for row in myreader:
    #        Mach.append(row[1])
    #        Aoa.append(row[0])

    # iterate on Mach number
    for i in range(len(MachList)):
        # local config and state
        konfig = copy.deepcopy(config)
        konfig.DISCARD_INFILES = 'YES'
        ztate  = copy.deepcopy(state)

        # set config options
        konfig.MACH_NUMBER = MachList[i]
        caseName='DIRECT_i_'+str(i)

        # run su2
        drag = SU2.eval.func('DRAG',konfig,ztate)
        lift = SU2.eval.func('LIFT',konfig,ztate)

        LiftList.append(lift)
        DragList.append(drag)

        # format output and write to file
        output = str(MachList[i])+", "\
        + str(lift) + ", " + str(drag) + "\n"
        f.write(output)

        # Store result in a subdirectory
        if os.path.isdir(caseName):
            os.system('rm -rf '+caseName)
        os.system('mv DIRECT '+caseName)

    # Close output file
    f.close()

    # plotting
    plt.figure()
    plt.plot( DragList,LiftList)
    plt.xlabel('Drag Coeffient')
    plt.ylabel('Lift Coefficient')
    plt.savefig('plot.png')
    #plt.show()
```

Initialize python objects that run SU2, modify options.

To modify config options from the python script:

```
config.EXT_ITER     = options.iterations
```

```
# imports
import numpy as np
import csv
import matplotlib.pyplot as plt
from optparse import OptionParser
import os, sys, shutil, copy, os.path
sys.path.append(os.environ['SU2_RUN'])
import SU2

def main():
# Command Line Options
    parser = OptionParser()
    parser.add_option("-f", "--file", dest="filename",
                      help="read config from FILE", metavar="FILE")
    parser.add_option("-n", "--partitions", dest="partitions", default=2,
                      help="number of PARTITIONS", metavar="PARTITIONS")
    parser.add_option("-i", "--iterations", dest="iterations", default=99999,
                      help="number of ITERATIONS", metavar="ITERATIONS")


    (options, args)=parser.parse_args()
    options.partitions = int( options.partitions )
    options.iterations = int( options.iterations )

    # load config, start state
    config = SU2.io.Config(options.filename)
    config.NUMBER_PART = options.partitions
    config.NZONES      = 1
    state  = SU2.io.State()

    # find solution files if they exist
    state.find_files(config)

    # prepare config
    config.NUMBER_PART = options.partitions
    config.EXT_ITER    = options.iterations

    # Initialize results arrays
    nMach = 5
    MachList=np.linspace(0.5,0.6,nMach)
    LiftList=[]
    DragList=[]

    # Output file
    f = open('output.csv', 'w')
    f.write(' Mach, CL, CD \n')

    #with open(options.traject) as csvfile:
    #    myreader = csv.reader(csvfile, delimiter=',')
    #    next(myreader, None)
    #    for row in myreader:
    #        Mach.append(row[1])
    #        Aoa.append(row[0])

    # iterate on Mach number
    for i in range(len(MachList)):
        # local config and state
        konfig = copy.deepcopy(config)
        konfig.DISCARD_INFILES = 'YES'
        ztate  = copy.deepcopy(state)

        # set config options
        konfig.MACH_NUMBER = MachList[i]
        caseName='DIRECT_i_'+str(i)

        # run su2
        drag = SU2.eval.func('DRAG',konfig,ztate)
        lift = SU2.eval.func('LIFT',konfig,ztate)

        LiftList.append(lift)
        DragList.append(drag)

        # format output and write to file
        output = str(MachList[i])+", "\
        + str(lift) + ", " + str(drag) + "\n"
        f.write(output)

        # Store result in a subdirectory
        if os.path.isdir(caseName):
            os.system('rm -rf '+caseName)
        os.system('mv DIRECT '+caseName)

    # Close output file
    f.close()

    # plotting
    plt.figure()
    plt.plot( DragList,LiftList)
    plt.xlabel('Drag Coeffient')
    plt.ylabel('Lift Coefficient')
    plt.savefig('plot.png')
    #plt.show()
```

Set up range of Mach numbers, open the output file, and start a for loop.
Note example for reading csv file.

```
# imports
import numpy as np
import csv
import matplotlib.pyplot as plt
from optparse import OptionParser
import os, sys, shutil, copy, os.path
sys.path.append(os.environ['SU2_RUN'])
import SU2

def main():
# Command Line Options
    parser = OptionParser()
    parser.add_option("-f", "--file", dest="filename",
                      help="read config from FILE", metavar="FILE")
    parser.add_option("-n", "--partitions", dest="partitions", default=2,
                      help="number of PARTITIONS", metavar="PARTITIONS")
    parser.add_option("-i", "--iterations", dest="iterations", default=99999,
                      help="number of ITERATIONS", metavar="ITERATIONS")


    (options, args)=parser.parse_args()
    options.partitions = int( options.partitions )
    options.iterations = int( options.iterations )

    # load config, start state
    config = SU2.io.Config(options.filename)
    config.NUMBER_PART = options.partitions
    config.NZONES      = 1
    state  = SU2.io.State()

    # find solution files if they exist
    state.find_files(config)

    # prepare config
    config.NUMBER_PART = options.partitions
    config.EXT_ITER    = options.iterations

    # Initialize results arrays
    nMach = 5
    MachList=np.linspace(0.5,0.6,nMach)
    LiftList=[]
    DragList=[]

    # Output file
    f = open('output.csv', 'w')
    f.write(' Mach, CL, CD \n')

    #with open(options.traject) as csvfile:
    #    myreader = csv.reader(csvfile, delimiter=',')
    #    next(myreader, None)
    #    for row in myreader:
    #        Mach.append(row[1])
    #        Aoa.append(row[0])

    # iterate on Mach number
    for i in range(len(MachList)):
        # local config and state
        konfig = copy.deepcopy(config)
        konfig.DISCARD_INFILES = 'YES'
        ztate  = copy.deepcopy(state)

        # set config options
        konfig.MACH_NUMBER = MachList[i]
        caseName='DIRECT_i_'+str(i)

        # run su2
        drag = SU2.eval.func('DRAG',konfig,ztate)
        lift = SU2.eval.func('LIFT',konfig,ztate)

        LiftList.append(lift)
        DragList.append(drag)

        # format output and write to file
        output = str(MachList[i])+", "\
        + str(lift) + ", " + str(drag) + "\n"
        f.write(output)

        # Store result in a subdirectory
        if os.path.isdir(caseName):
            os.system('rm -rf '+caseName)
        os.system('mv DIRECT '+caseName)

    # Close output file
    f.close()

    # plotting
    plt.figure()
    plt.plot( DragList,LiftList)
    plt.xlabel('Drag Coeffient')
    plt.ylabel('Lift Coefficient')
    plt.savefig('plot.png')
    #plt.show()
```

Copy the config and state objects using copy.deepcopy and set the Mach number

```
# imports
import numpy as np
import csv
import matplotlib.pyplot as plt
from optparse import OptionParser
import os, sys, shutil, copy, os.path
sys.path.append(os.environ['SU2_RUN'])
import SU2

def main():
# Command Line Options
    parser = OptionParser()
    parser.add_option("-f", "--file", dest="filename",
                      help="read config from FILE", metavar="FILE")
    parser.add_option("-n", "--partitions", dest="partitions", default=2,
                      help="number of PARTITIONS", metavar="PARTITIONS")
    parser.add_option("-i", "--iterations", dest="iterations", default=99999,
                      help="number of ITERATIONS", metavar="ITERATIONS")


    (options, args)=parser.parse_args()
    options.partitions = int( options.partitions )
    options.iterations = int( options.iterations )

    # load config, start state
    config = SU2.io.Config(options.filename)
    config.NUMBER_PART = options.partitions
    config.NZONES      = 1
    state  = SU2.io.State()

    # find solution files if they exist
    state.find_files(config)

    # prepare config
    config.NUMBER_PART = options.partitions
    config.EXT_ITER    = options.iterations

    # Initialize results arrays
    nMach = 5
    MachList=np.linspace(0.5,0.6,nMach)
    LiftList=[]
    DragList=[]

    # Output file
    f = open('output.csv', 'w')
    f.write(' Mach, CL, CD \n')

    #with open(options.traject) as csvfile:
    #    myreader = csv.reader(csvfile, delimiter=',')
    #    next(myreader, None)
    #    for row in myreader:
    #        Mach.append(row[1])
    #        Aoa.append(row[0])

    # iterate on Mach number
    for i in range(len(MachList)):
        # local config and state
        konfig = copy.deepcopy(config)
        konfig.DISCARD_INFILES = 'YES'
        ztate  = copy.deepcopy(state)

        # set config options
        konfig.MACH_NUMBER = MachList[i]
        caseName='DIRECT_i_'+str(i)

        # run su2
        drag = SU2.eval.func('DRAG',konfig,ztate)
        lift = SU2.eval.func('LIFT',konfig,ztate)

        LiftList.append(lift)
        DragList.append(drag)

        # format output and write to file
        output = str(MachList[i])+", "\
        + str(lift) + ", " + str(drag) + "\n"
        f.write(output)

        # Store result in a subdirectory
        if os.path.isdir(caseName):
            os.system('rm -rf '+caseName)
        os.system('mv DIRECT '+caseName)

    # Close output file
    f.close()

    # plotting
    plt.figure()
    plt.plot( DragList,LiftList)
    plt.xlabel('Drag Coeffient')
    plt.ylabel('Lift Coefficient')
    plt.savefig('plot.png')
    #plt.show()
```

Evaluate lift and drag using SU2.eval.func('DRAG',konfig,ztate)

Note that only the first 'eval' runs SU2_CFD, after that the output values are stored in the state object.

```
# imports
import numpy as np
import csv
import matplotlib.pyplot as plt
from optparse import OptionParser
import os, sys, shutil, copy, os.path
sys.path.append(os.environ['SU2_RUN'])
import SU2

def main():
# Command Line Options
    parser = OptionParser()
    parser.add_option("-f", "--file", dest="filename",
                      help="read config from FILE", metavar="FILE")
    parser.add_option("-n", "--partitions", dest="partitions", default=2,
                      help="number of PARTITIONS", metavar="PARTITIONS")
    parser.add_option("-i", "--iterations", dest="iterations", default=99999,
                      help="number of ITERATIONS", metavar="ITERATIONS")


    (options, args)=parser.parse_args()
    options.partitions = int( options.partitions )
    options.iterations = int( options.iterations )

    # load config, start state
    config = SU2.io.Config(options.filename)
    config.NUMBER_PART = options.partitions
    config.NZONES      = 1
    state  = SU2.io.State()

    # find solution files if they exist
    state.find_files(config)

    # prepare config
    config.NUMBER_PART = options.partitions
    config.EXT_ITER    = options.iterations

    # Initialize results arrays
    nMach = 5
    MachList=np.linspace(0.5,0.6,nMach)
    LiftList=[]
    DragList=[]

    # Output file
    f = open('output.csv', 'w')
    f.write(' Mach, CL, CD \n')

    #with open(options.traject) as csvfile:
    #    myreader = csv.reader(csvfile, delimiter=',')
    #    next(myreader, None)
    #    for row in myreader:
    #        Mach.append(row[1])
    #        Aoa.append(row[0])

    # iterate on Mach number
    for i in range(len(MachList)):
        # local config and state
        konfig = copy.deepcopy(config)
        konfig.DISCARD_INFILES = 'YES'
        ztate  = copy.deepcopy(state)

        # set config options
        konfig.MACH_NUMBER = MachList[i]
        caseName='DIRECT_'+str(i)

        # run su2
        drag = SU2.eval.func('DRAG',konfig,ztate)
        lift = SU2.eval.func('LIFT',konfig,ztate)

        LiftList.append(lift)
        DragList.append(drag)

        # format output and write to file
        output = str(MachList[i])+", "\
        + str(lift) + ", " + str(drag) + "\n"
        f.write(output)

        # Store result in a subdirectory
        if os.path.isdir(caseName):
            os.system('rm -rf '+caseName)
        os.system('mv DIRECT '+caseName)

    # Close output file
    f.close()

    # plotting
    plt.figure()
    plt.plot( DragList,LiftList)
    plt.xlabel('Drag Coeffient')
    plt.ylabel('Lift Coefficient')
    plt.savefig('plot.png')
    #plt.show()
```

Store results, close output file, and plot results

```python
# imports
import numpy as np
import csv
import matplotlib.pyplot as plt
from optparse import OptionParser
import os, sys, shutil, copy, os.path
sys.path.append(os.environ['SU2_RUN'])
import SU2

def main():
# Command Line Options
    parser = OptionParser()
    parser.add_option("-f", "--file", dest="filename",
                      help="read config from FILE", metavar="FILE")
    parser.add_option("-n", "--partitions", dest="partitions", default=2,
                      help="number of PARTITIONS", metavar="PARTITIONS")
    parser.add_option("-i", "--iterations", dest="iterations", default=99999,
                      help="number of ITERATIONS", metavar="ITERATIONS")


    (options, args)=parser.parse_args()
    options.partitions = int( options.partitions )
    options.iterations = int( options.iterations )

    # load config, start state
    config = SU2.io.Config(options.filename)
    config.NUMBER_PART = options.partitions
    config.NZONES      = 1
    state  = SU2.io.State()

    # find solution files if they exist
    state.find_files(config)

    # prepare config
    config.NUMBER_PART = options.partitions
    config.EXT_ITER    = options.iterations

    # Initialize results arrays
    nMach = 5
    MachList=np.linspace(0.5,0.6,nMach)
    LiftList=[]
    DragList=[]

    # Output file
    f = open('output.csv', 'w')
    f.write(' Mach, CL, CD \n')

    #with open(options.traject) as csvfile:
    #    myreader = csv.reader(csvfile, delimiter=',')
    #    next(myreader, None)
    #    for row in myreader:
    #        Mach.append(row[1])
    #        Aoa.append(row[0])

    # iterate on Mach number
    for i in range(len(MachList)):
        # local config and state
        konfig = copy.deepcopy(config)
        konfig.DISCARD_INFILES = 'YES'
        ztate  = copy.deepcopy(state)

        # set config options
        konfig.MACH_NUMBER = MachList[i]
        caseName='DIRECT_i_'+str(i)

        # run su2
        drag = SU2.eval.func('DRAG',konfig,ztate)
        lift = SU2.eval.func('LIFT',konfig,ztate)

        LiftList.append(lift)
        DragList.append(drag)

        # format output and write to file
        output = str(MachList[i])+", "\
        + str(lift) + ", " + str(drag) + "\n"
        f.write(output)

        # Store result in a subdirectory
        if os.path.isdir(caseName):
            os.system('rm -rf '+caseName)
        os.system('mv DIRECT '+caseName)

    # Close output file
    f.close()

    # plotting
    plt.figure()
    plt.plot( DragList,LiftList)
    plt.xlabel('Drag Coeffient')
    plt.ylabel('Lift Coefficient')
    plt.savefig('plot.png')
    #plt.show()
```

Resources

- Use interactive mode to test commands.
- Plotting with python: http://matplotlib.org/faq/howto_faq.html
- Numpy (arrays, random numbers, etc): https://docs.scipy.org/doc/numpy/reference/index.html
- File I/O with Python: https://docs.python.org/2/tutorial/inputoutput.html#reading-and-writing-files
- CSV file reader/writer: https://docs.python.org/2/library/csv.html
- Output function names: SU2_PY/SU2/io/tools.py → get_headerMap()