Perspectives on Computational Analysis

Problem Set 7

Problem 1

Code with Description of Answers

```python
import problem1

def test_factor():
    assert problem1.smallest_factor(1)==1, "failed on 1"
    assert problem1.smallest_factor(2)==2, "failed on even prime number"
    assert problem1.smallest_factor(3)==3, "failed on smallest odd prime number"
    assert problem1.smallest_factor(4)==2, "failed on smallest square number other than one"
    assert problem1.smallest_factor(6)==2, "failed with factor of two prime factors, of which one is even"
    assert problem1.smallest_factor(15)==3, "failed with factor of two prime factors, both of which are odd"
```

Prob1: (see code in test_prob1.py)

On running the tests, answers initially seem to come out correctly for 1,2, and 3. Errors begin to occur at 4 (the answer comes out as 4, instead of 2). Initially, I checked if this was a problem with perfect squares. The square root of 9 and larger numbers did not cause any such problems. I also checked for different combinations of odd and even factors. Problems occurred only for 4. On a closer inspection of the code, it became clear that the issue was a particularity of the range expression in Python, where the second argument is not included in the iteration. For the edge case of 4, the range would thus become range(2,2), which would not produce any meaningful output. This would lead the programme to simply execute the last line, and return 4 itself. To remedy this, I added '+1' to the second argument of the range expression. After this modification, the code passed all tests.

Prob 2: (see code in test_prob2.py)

```python
import problem2
def test_month():
    assert problem2.month_length("June")==30, "Wrong number of days for a month with 30 days"
    assert problem2.month_length("November")==30, "Wrong number of days for a month with 30 days"
    assert problem2.month_length("December")==31, "Wrong number of days for a month with 31 days"
    assert problem2.month_length("May")==31, "Wrong number of days for a month with 31 days"
    assert problem2.month_length("February")==28, "Leap year was not specified, but calculation was done on the basis of leap years"
    assert problem2.month_length("February", leap_year=True)==29, "Leap year was specified, but calculation was done on the basis of non-leap years"
    assert problem2.month_length(None)==None, "Blank response did not generate None type"
    assert  problem2.month_length("june")==None, "Incorrect capitalization was treated as correct spelling"
    assert  problem2.month_length("Septembre")==None, "Incorrect spelling was treated as correct spelling"
```

The unit tests check for matching the correct month with the correct number of days (either 31 or 30, based on the dictionaries provided). For this purpose, I tested the code's results against 'December' and 'June' respectively from the two groups. In the case of February, we check for the correct of 29 or 28 days given that the year is a leap year or not respectively. Finally, the code also ensures that incorrect spellings of the months or an empty argument are not processed, and simply return 'None' as the output.

Prob 3: (see code in test_prob3.py)

```python
import problem2
def test_month():
    assert problem2.month_length("June")==30, "Wrong number of days for a month with 30 days"
    assert problem2.month_length("November")==30, "Wrong number of days for a month with 30 days"
    assert problem2.month_length("December")==31, "Wrong number of days for a month with 31 days"
    assert problem2.month_length("May")==31, "Wrong number of days for a month with 31 days"
    assert problem2.month_length("February")==28, "Leap year was not specified, but calculation was done on the basis of leap years"
    assert problem2.month_length("February", leap_year=True)==29, "Leap year was specified, but calculation was done on the basis of non-leap years"
    assert problem2.month_length(None)==None, "Blank response did not generate None type"
    assert  problem2.month_length("june")==None, "Incorrect capitalization was treated as correct spelling"
    assert  problem2.month_length("Septembre")==None, "Incorrect spelling was treated as correct spelling"
```

The code first ensures that the third argument of the function is of type string. In case that is not true, the pytest.raises method ensures that if a TypeError is based on the written raise statement. The code then tests out each of the first three operations- addition, multiplication and subtraction on the numbers 3 and 4. The case of division gets special attention. First, we check for 0 in the denominator. The pytest.raises method ensures that if a ZeroDivisionError is based on the written raise statement. Secondly, we ensure that the division operator works correctly for both float and integer results. Finally, we use pytest.raises again- this time with the ValueError – to ensure that our raise statement comes into play whenever any operator other than the ones designated in the codes is used.

Problem 2

Based on the results of the PyTest commands, we see that the function has passed all the required tests.

----------- coverage: platform win32, python 3.6.5-final-0 -----------

Name   Stmts   Miss Cover

----

get_r.py   5   0   100%

```
test_r.py     29     0   100%
-------------------------------
TOTAL         34     0   100%
```

Problem 3

a) When it was first introduced in the 1960s, Rational Choice Theory faced two primary criticisms. The first applied ex-ante at the level of implausible model assumptions (regarding the preferences, computation capabilities or knowledge of social actors). The second related ex-post to the predictions generated by such models. Both the assumptions and predictions were considered to not match empirical observation.

b) Mental simulation in everyday life tends to be applied both ex-ante (to predict the future behaviour of actors) and ex-post (to rationalize the observed past behaviour of actors through inferences on either the actors themselves or the situational context they were placed in). The author adds that human beings switch between these two modes frequently and unconsciously. The author concedes that this modus operandi- which he refers to as 'commonsense theories of action'- often prove useful in daily situations. However, their validity in those narrow contexts can mislead social scientists to erroneously consider them universally valid.

   This 'understandability' of an explanation for an observed phenomenon (making 'sense' intuitively) cannot serve as an epistemic substitute for causality (claims on the specific or even generalizable causal mechanism). Moreover, seemingly important mechanisms- when assessed ex-ante- could well produce incorrect predictions. Conversely, predictably accurate mechanisms- when assessed ex-post- could not even have been reliably known ex-ante. While such errors in daily situations are either minor or swiftly fixed through feedback mechanisms, they can prove far more lethal if unconsciously embedded and remaining unquestioned within social science literature.

   Thus, Watt's primary concern is that reliance on commonsense theories of action (via mental simulation) due to their narrow scope and fallibility cannot serve as the foundation for developing sociological theories that are broadly applicable and would produce scientifically valid explanations.

c) Watts concedes that no universal solution exists to the problems he outlined with respect to rational theory and causal mechanisms. However, he asserts that several existing partial solutions may be harnessed. Moreover, all of them address Woodward's (2003) 'what-if-it-had-been-different' question while also presenting distinct standards of evidence to justify these claims.

His solution consists of three parts. Firstly, he advocates experimental methods in sociology to the extent possible (in order of preference- field experiments, natural experiments and quasi-experiments, and finally, lab experiments).However, these suffer on the grounds of generalizability and successfully studying collective entities. Next, he suggests non-experimental counterfactual methods (based on statistical modelling), preferably with observational studies with large sample sizes. Finally, he suggests that explanatory hypotheses be evaluated by their predictive abilities largely through out-of-sample testing, which would facilitate not only individual probabilistic predictions, but also broader predictions on outcome patterns and stylized facts. Prediction, if thus defined, would allow for rigorously testing of a variety of methods and types of studies.

d) A model, by definition, aims to isolate or focus on only one aspect of an observed phenomenon, and accordingly ignores or sidelines others- thereby separating the 'signal' of importance from the noise (Silver, 2012). Theory thus helps not only in determining which data to focus on, but establishes a hierarchy of salience within that data of the relationships between specific variables- a critical concern in the digital age of massive datasets. This helps researchers identify and fine-tune theories on causal mechanisms.

Furthermore, models help make assumptions on these mechanisms explicit. For example, Keane (2010) develops the example of draft lotteries as natural experiments (cited in this paper) and highlights how even such 'ideal instruments' provide no information on a priori assumptions. Imbens and Angrist's (1994) subsequent research on conscription for the Vietnam War found heterogeneous effects of service on veterans' earnings, and a general decline- but without establishing any chain of causality. Keane (2010) asserts that a priori assumptions would be essential to derive any meaningful insights from data beyond the most basic descriptive statistics. Theoretical models merely spell out these assumptions explicitly, thus setting the stage for inference and allowing for disproving and revision of theories in case found to be untenable for the research questions and causal mechanisms they seek to address.

REFERENCES

Keane, M (2010), '*Structural vs. atheoretic approaches to econometrics*' Journal of Econometrics 156 (2010) 3–20

Imbens, G., Angrist, J., (1994). '*Identification and estimation of local average treatment effects*.' Econometrica 62, 467–476.

Silver, N. (2012). *The signal and the noise: Why so many predictions fail--but some don't*. New York: Penguin Press.

Woodward, J. (2003) '*Making Things Happen*: A Theory of Causal Explanation,' Oxford University Press