1.3 Exercises NS-2

### Topic of this homework:
Prime numbers, greatest common divisors, the continued fraction algorithm

Deliverable: Answers to questions.

### Prime numbers

**Problem # 1:** Every integer may be written as a product of primes.

---

**Q 1.1:** Put the numbers 1,000,000, 1,000,004 and 999,999 in the form \( N = \prod_k \pi_k^{\beta_k} \) (Hint: Use Matlab/Octave to find the prime factors).

**Sol:**
- \( 1,000,000 = 2^6 \cdot 5^6 \)
- \( 1,000,004 = 2^2 \cdot 53^2 \cdot 89 \)
- \( 999,999 = 3^3 \cdot 7 \cdot 11 \cdot 13 \cdot 37 \)

**Q 1.2:** Give a generalized formula for the natural logarithm of a number, \( \ln(N) \), in terms of its primes \( \pi_k \) and their multiplicities \( \beta_k \). Express your answer as a sum of terms.

**Sol:** \( \ln N = \sum_k \beta_k \ln(\pi_k) \)

### Problem # 2: Using the computer

**Q 2.1:** Explain why the following brief Matlab/Octave program returns the prime numbers \( \pi_k \) between 1 and 100.

```matlab
n=2:100;
k = isprime(n);
n(k)
```

**Sol:** The first line \( n = 2 : 100 \) defines the row vector \( n = [2, 3, 4, \ldots, 100] \). The second line creates a row vector the same length as \( n \), with entries of 1 if the element is prime and zero if the element is not prime. The third line \( n(k) \) prints out \( n() \) if \( k = 1 \), namely it is a list of all the primes from 2 to 100. Run this program without the ';' at the end of each line, and to see what it is doing.

**Q 2.2:** How many primes are there between 2 and \( N = 100 \)?

**Sol:** \( \text{length}(n(k)) \) returns 25. Thus there are 25 primes less than 100 (\( N/4 \), on average).

### Problem # 3: Prime numbers may be identified using a `sieve`.

**Q 3.1:** By hand, perform the sieve of Eratosthenes for \( n = 1 \ldots 49 \). Circle each prime \( p \), then cross out each number which is a multiple of \( p \).

**Sol:** Note: 1 should not be circled as it is not a prime.
Q 3.2: What is the largest number you need to consider before only primes remain?
Sol: $\lfloor \sqrt{50} \rfloor = \lfloor 7.0711 \rfloor = 7.$

Q 3.3: Generalize: for $n = 1 \ldots N$, what is the highest number you need to consider before only the primes remain?
Sol: $\text{floor}(\sqrt{N})$

Q 3.4: Write each of these numbers as a product of primes:
$22 = 2 \cdot 11 = \pi_1 \pi_5$
$30 = 2 \cdot 3 \cdot 5 = \pi_1 \pi_2 \pi_3$
$34 = 2 \cdot 17 = \pi_1 \pi_7$
$43 = \pi_{14}$
$44 = 4 \cdot 11 = \pi_2^2 \pi_5$
$48 = 4 \cdot 12 = 4^2 \cdot 3 = \pi_1^4 \pi_2$
$49 = 7^2 = \pi_4$

Q 3.5: Find the largest prime $\pi_k \leq 100$? Hint: Do not use Matlab/Octave other than to check your answer. Hint: Write out the numbers starting with 100 and counting backwards: 100, 99, 98, 97, $\cdots$. Cross off the even numbers, leaving 99, 97, 95, $\cdots$. Pull out a factor (only 1 is necessary to show that it is not prime).
Sol: $99=11 \cdot 9, \pi_{25} = 97$.

Q 3.6: Find the largest prime $\pi_k \leq 1000$? Hint: Do not use Matlab/Octave other than to check your answer.
Sol: Write out the numbers starting with 1000 and counting backwards: 1000, 999, 998, 997, $\cdots$. Cross off the even numbers, leaving 999, 997, 995, $\cdots$. Pull out a factor (only 1 is necessary to show that it is not prime). $9 \cdot 111, 997 = \pi_{168}, 5 \cdot 199 = \pi_3 \cdot \pi_{46}$.

Q 3.7: Explain why $\pi_k^{-s} = e^{-s \ln \pi_k}$.
Sol: This follows from the identity $z^a = e^{a \ln z}$ with $a, z \in \mathbb{C}$.

Greatest common divisors
Consider the Euclidean algorithm to find the greatest common divisor (GCD; the largest common prime factor) of two numbers. Note this algorithm may be performed using one of two methods:

<table>
<thead>
<tr>
<th>Method</th>
<th>Division</th>
<th>Subtraction</th>
</tr>
</thead>
<tbody>
<tr>
<td>On each iteration...</td>
<td>$a_{i+1} = b_i$</td>
<td>$a_{i+1} = \max(a_i, b_i) - \min(a_i, b_i)$</td>
</tr>
<tr>
<td></td>
<td>$b_{i+1} = a_i - b_i \cdot \text{floor}(a_i/b_i)$</td>
<td>$b_{i+1} = \min(a_i, b_i)$</td>
</tr>
<tr>
<td>Terminates when...</td>
<td>$b = 0 \ (\gcd = a)$</td>
<td>$b = 0 \ (\gcd = a)$</td>
</tr>
</tbody>
</table>

The division method (Eq. 2.1, Sect. 2.1.2, Lec 5, Ch. 2) is preferred because the subtraction method is much slower.

Problem # 4: Understand the Euclidean (GCD) algorithm
–Q 4.1: Use the Octave/Matlab command `factor` to find the prime factors of $a = 85$ and $b = 15$.

**Sol:** From Octave’s `factor()` we find $85 = 17 \cdot 5$, $15 = 3 \cdot 5$.

–Q 4.2: What is the greatest common prime factor of these two numbers?

**Sol:** The largest common factor $\gcd(85, 15)$ is 5.

–Q 4.3: By hand, perform the Euclidean algorithm for $a = 85$ and $b = 15$.

**Sol:** Division method:

\[
\begin{align*}
a_1 &= 15 & b_1 &= 85 - 15 \div 15 \cdot 75 = 10 \\
\end{align*}
\]

\[
\begin{align*}
a_2 &= 10 & b_2 &= 15 - 10 \div 15 \cdot 10 = 5 \\
\end{align*}
\]

\[
\begin{align*}
a_3 &= 5 & b_3 &= 10 - 5 \div 3 \cdot 5 = 0 \\
\end{align*}
\]

\[
\therefore \ gcd = 5
\]

Subtraction method:

\[
\begin{align*}
a_1 &= 85 - 15 = 70 & b_1 &= 15 \\
\end{align*}
\]

\[
\begin{align*}
a_2 &= 70 - 15 = 55 & b_2 &= 15 \\
\end{align*}
\]

\[
\begin{align*}
a_3 &= 55 - 15 = 40 & b_3 &= 15 \\
\end{align*}
\]

\[
\begin{align*}
a_4 &= 40 - 15 = 25 & b_4 &= 15 \\
\end{align*}
\]

\[
\begin{align*}
a_5 &= 25 - 15 = 10 & b_5 &= 15 \\
\end{align*}
\]

\[
\text{swap}
\]

\[
\begin{align*}
a_6 &= 15 - 10 = 5 & b_6 &= 10 \\
\end{align*}
\]

\[
\begin{align*}
a_7 &= 10 - 5 = 5 & b_7 &= 5 \\
\end{align*}
\]

\[
\therefore \ gcd = 5
\]

–Q 4.4: By hand, perform the Euclidean algorithm for $a = 75$ and $b = 25$. Is the result a prime number?

**Sol:** Division method:

\[
\begin{align*}
a_1 &= 25 & b_1 &= 75 - 25 \div 75 \cdot 3 = 0 \\
\end{align*}
\]

Subtraction method:

\[
\begin{align*}
a_1 &= 75 - 25 = 50 & b_1 &= 25 \\
\end{align*}
\]

\[
\begin{align*}
a_2 &= 50 - 25 = 25 & b_2 &= 25 \\
\end{align*}
\]

\[
\therefore \ gcd = 25
\]

The result is $25 = 5^2$, the square of a prime number.

–Q 4.5: Consider the first step of the GCD division algorithm when $a < b$ (e.g. $a = 25$ and $b = 75$). What happens to $a$ and $b$ in the first step? Does it matter if you begin the algorithm with $a < b$ vs. $b < a$?

**Sol:** If $a < b$, the first step of the division algorithm swaps the terms ($a \rightarrow b$ and $b \rightarrow a$).
Q 4.6: Describe in your own words how the GCD algorithm works. Try the algorithm using numbers which have already been separated into factors (e.g. \( a = 5 \cdot 3 \) and \( b = 7 \cdot 3 \)).

**Sol:** Division method:

\[
\begin{align*}
a_1 &= 5 \cdot 3 & b_1 &= 7 \cdot 3 - 5 \cdot 3 \left\lfloor \frac{7 \cdot 3}{5 \cdot 3} \right\rfloor = 2 \cdot 3 \\
a_2 &= 2 \cdot 3 & b_2 &= 5 \cdot 3 - 2 \cdot 3 \left\lfloor \frac{5 \cdot 3}{2 \cdot 3} \right\rfloor = 1 \cdot 3 \\
a_3 &= 1 \cdot 3 & b_3 &= 2 \cdot 3 - 1 \cdot 3 \left\lfloor \frac{2 \cdot 3}{1 \cdot 3} \right\rfloor = 0
\end{align*}
\]

**Subtraction method:**

\[
\begin{align*}
a_1 &= 7 \cdot 3 - 5 \cdot 3 = 2 \cdot 3 & b_1 &= 5 \cdot 3 \\
a_2 &= 5 \cdot 3 - 2 \cdot 3 = 3 \cdot 3 & b_2 &= 2 \cdot 3 \\
a_3 &= 3 \cdot 3 - 2 \cdot 3 = 1 \cdot 3 & b_3 &= 2 \cdot 3 \\
a_4 &= 2 \cdot 3 - 1 \cdot 3 = 1 \cdot 3 & b_4 &= 1 \cdot 3
\end{align*}
\]

The algorithm iteratively converges on the GCD by subtracting out multiples of the GCD until only the GCD is left.

**Problem # 5: Coprimes**

1. Define the term **coprime**. **Sol:** when two integers have no common factors they are said to be **coprime**

2. How can the Euclidean algorithm be used to identify coprimes? **Sol:** If \( \gcd(a, b) = 1 \) they only have 1 as a common factor, thus they are coprime.

3. Give at least one application of the Euclidean algorithm. **Sol:** Given two integers \( n, d \in \mathbb{Z} \), if we wish to reduce the fraction \( n/d \), we must cancel the common factors. Example: If \( n = 9, d = 6 \) then \( 9/6 = (3 \cdot 3)/(2 \cdot 3) = 3/2 \), where the GCD, 3, may be identified using the Euclidean algorithm. While this fraction may be easily simplified via inspection, the GCD algorithm could be very helpful for larger numbers \( n, d \).

Q 5.1: Write a Matlab function, \( \text{function } x = \text{my\_gcd}(a,b) \), which uses the Euclidean algorithm to find the GCD of any two inputs \( a \) and \( b \). Test your function on the \((a,b)\) combinations from parts (a) and (b). Include a printout (or handwrite) your algorithm to turn in.

**Hints and advice:**

- Don’t give your variables the same names as Matlab functions! Since \( \gcd \) is an existing Matlab/Octave function, if you use it as a variable or function name, you won’t be able to use \( \text{gcd}() \) to check your \( \text{gcd}() \) function. Try \( \text{clear all} \) to recover from this problem.

- Try using a ‘while’ loop for this exercise (see Matlab documentation for help).

- You may need to make some temporary variables for \( a \) and \( b \) in order to perform the algorithm.

**Sol:** Division method:

\[
\begin{align*}
\text{function } x &= \text{my\_gcd}(a,b) \\
\text{while } b>0 \\
\text{atmp} &= a; \text{btmp} = b; \\
a &= \text{btmp}; \text{b} &= \text{atmp} - \text{btmp} \times \text{floor(atmp/btmp)}; \\
\text{end}
\end{align*}
\]
Subtraction method:

```matlab
function x = my_gcd(a,b)
while a~==b
    atmp = a; btmp = b;
    a = max(atmp,btmp) - min(atmp,btmp); b = min(atmp,btmp);
end
```

Algebraic generalization of the GCD (Euclidean) algorithm

**Problem # 6:** In this problem we are looking for integer solutions \((m, n) \in \mathbb{Z}\) to the equations \(ma + nb = \gcd(a, b)\) and \(ma + nb = 0\) given positive integers \((a, b) \in \mathbb{Z}^+\).

Note that this requires that either \(m\) or \(n\) be negative. These solutions may be found using the Euclidean algorithm only if \((a, b)\) are coprime \((a \perp b)\). Note that integer (whole number) polynomial relations such as these are known as ‘Diophantine equations.’ The above equations are linear Diophantine equations, possibly the simplest form of such relations.

**Example:** \(\gcd(2, 3)=1\): For \((a, b) = (2, 3)\), the result is as follows:

\[
\begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ 3 & -2 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \end{bmatrix}.
\]

Thus from the above equation we find the solution \((m, n)\) to the integer equation

\[2m + 3n = \gcd(2, 3) = 1,
\]

namely \((m, n) = (-1, 1)\) (i.e., \(-2+3 = 1\)). There is also a second solution \((3, -2)\) (i.e., \(3\cdot2-2\cdot3 = 0\)), which represents the terminating condition. Thus these two solutions are a pair and the solution only exists if \((a, b)\) are coprime \((a \perp b)\).

Subtraction method: This method is more complicated than the division algorithm, because at each stage we must check if \(a < b\). Define

\[
\begin{bmatrix} a_0 \\ b_0 \end{bmatrix} = \begin{bmatrix} a \\ b \end{bmatrix}, \quad Q = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}, \quad S = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}
\]

where \(Q\) sets \(a_{i+1} = a_i - b_i\) and \(b_{i+1} = b_i\) assuming \(a_i > b_i\), and \(S\) is a ‘swap-matrix’ which swaps \(a_i\) and \(b_i\) if \(a_i < b_i\). Using these matrices, the algorithm is implemented by assigning

\[
\begin{bmatrix} a_{i+1} \\ b_{i+1} \end{bmatrix} = Q \begin{bmatrix} a_i \\ b_i \end{bmatrix} \text{ for } a_i > b_i, \quad \begin{bmatrix} a_{i+1} \\ b_{i+1} \end{bmatrix} = QS \begin{bmatrix} a_i \\ b_i \end{bmatrix} \text{ for } a_i < b_i.
\]

The result of this method is a cascade of \(Q\) and \(S\) matrices. For \((a, b) = (2, 3)\), the result is as follows:

\[
\begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \end{bmatrix} = \begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \end{bmatrix}.
\]

Thus we find two solutions \((m, n)\) to the integer equation \(2m + 3n = \gcd(2, 3) = 1\).

---

**Q 6.1:** By inspection, find at least one integer pair \((m, n)\) that satisfies \(12m + 15n = 3\).

**Sol:** By inspection, \((m, n) = (-1, 1)\) is one solution.
Using matrix methods for the Euclidean algorithm, find integer pairs \((m, n)\) that satisfy \(12m + 15n = 3\) and \(12m + 15n = 0\). Show your work!!!

**Sol:** Division method:

\[
\begin{bmatrix} 3 \\ 0 \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ 5 & -4 \end{bmatrix} \begin{bmatrix} 12 \\ 15 \end{bmatrix}
\]

Subtraction method:

\[
\begin{bmatrix} 3 \\ 0 \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ 4 & -3 \end{bmatrix} \begin{bmatrix} 12 \\ 15 \end{bmatrix}
\]

**Q 6.2:** Does the equation \(12m + 15n = 1\) have integer solutions for \(n\) and \(m\)? Why, or why not?

**Sol:** No, because gcd\((12, 15) = \text{gcd}(3 \times 4, 3 \times 5) = 3\), not 1. Thus there are no Diophantine solutions to this equation.

**Problem # 7: Matrix approach:**

It can be difficult to keep track of the a’s and b’s when the algorithm has many steps. We need an alternative way to run the Euclidean algorithm, using matrix algebra. Matrix methods provide a more transparent approach to the operations on \((a, b)\). Thus the Euclidean algorithm can be classified in terms of standard matrix operations.

**Q 7.1:** Write out the matrix approach, discussed at the end of Sect. 2.4.1 (Eq. 2.8, p. 55).

**Sol:** Division method:

Define

\[
\begin{bmatrix} a \\ b \end{bmatrix}_0 = \begin{bmatrix} a_0 \\ b_0 \end{bmatrix}, \quad \begin{bmatrix} a \\ b \end{bmatrix}_{i+1} = \begin{bmatrix} 0 & 1 \\ 1 & -\lfloor a/b \rfloor \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix}_i
\]

**Continued fractions**

**Problem # 8:** Here we explore the continued fraction algorithm (CFA), Sect. 2.4.2, (p. 56).

In its simplest form the CFA starts with a real number, which we denote as \(\alpha \in \mathbb{R}\). Let us work with an irrational real number, \(\pi \in \mathbb{I}\), as an example, because its CFA representation will be infinitely long. We can represent the CFA coefficients \(\alpha\) as a vector of integers \(n_k, k = 1, 2 \cdots \infty\)

\[
\alpha = [n_1; n_2, n_3, n_4, \cdots]
\]

\[
= n_1 + \frac{1}{n_2 + \frac{1}{n_3 + \frac{1}{n_4 + \cdots}}}
\]

As discussed in Section 2.4.1 (p. 52), the CFA is recursive, with three steps per iteration:

For \(\alpha_1 = \pi, n_1 = 3, r_1 = \pi - 3\) and \(\alpha_2 = 1/r_1\).

\[
\alpha_2 = 1/0.1416 = 7.0625 \cdots
\]

\[
\alpha_1 = n_1 + \frac{1}{\alpha_2} = n_1 + \frac{1}{n_2 + \frac{1}{n_3 + \cdots}}
\]

In terms of a Matlab/Octave script

```matlab
alpha0 = pi;
K=10;
n=zeros(1,K); alpha=zeros(1,K);
```
alpha(1)=alpha0;

for k=2:K %k=1 to K
n(k)=round(alpha(k-1));
%
%disp([fix(k), round(n(k)), alpha(k)]); pause(1)
end

%Now compair this to matlab's rat() function
rat(alpha0,1e-20)

–Q 8.1: By hand (you may use Matlab/Octave as a calculator), find the first 3 values of \( n_k \)
for \( \alpha = e^\pi \).

**Sol:** The CFA for this is: 
\[
e^\pi = 23.1407 \cdots = [23; 7, 9, 4, \cdots].
\]

–Q 8.2: For part (1), what is the error (remainder) when you truncate the continued fraction
after \( n_1, \ldots, n_3 \)? Give the absolute value of the error, and the percentage error relative to the
original \( \alpha \).

**Sol:** The remainder is 
\[
e^\pi - (23 + 1/(7 + (1/9)))
\]
which gives an error of 
\[
|e^\pi - (23 + 1/(7 + (1/9)))|/e^\pi = 2.92 \cdot 10^{-6} = 0.0003\%
\]

–Q 8.3: Use the Matlab/Octave program provided to find the first 10 values of \( n_k \) for
\( \alpha = e^\pi \), and verify your result using the Matlab/Octave command \texttt{rat()}.

**Sol:** \( e^\pi = 23.1407 \cdots = [23; 7, 9, 4, -2, -591, -2, -10, 3, -2, \cdots] \).

–Q 8.4: Discuss the similarities and differences between the Euclidian algorithm (EA) and
CFA.

**Sol:**

1. Both are recursive, meaning that the steps are repeated one after another.

2. The EA starts from two numbers (a,b). The output of the \texttt{gcd(a,b)} is the GCD. The CFA starts with
a single number and the output is a sequence of integers. If the sequence terminates the number
was rational. If the sequence does not terminate, the number is irrational.

3. The EA works with the difference between the minimum and maximum of the two numbers
whereas the CFA works with the rounding function and the reciprocal of the error.

4. It would seem that the goals of the two algorithms, the starting point, and the results are totally
different. Both are very useful and powerful. Both generalize to more difficult situations than
working with simple numbers.

–Q 8.5: Extra Credit Show that the CFA is the inverse operation (i.e., the CFA is the GCD,
run in reverse) (Hint: see Sect. 2.4.1 (p. 52)). **NOT ON EXAM**

The full solution to this problem needs further analysis. **Sol:**

Starting from Eq. 2.7
\[
\begin{bmatrix}
0 & 1 \\
1 & 0
\end{bmatrix}
\begin{bmatrix}
1 & -1 \\
0 & 1
\end{bmatrix}
\left[\frac{m}{n}\right] =
\begin{bmatrix}
0 & 1 \\
1 & 0
\end{bmatrix}
\begin{bmatrix}
1 & -1 \\
0 & 1
\end{bmatrix}
\frac{m}{n} + 1 =
\begin{bmatrix}
0 & 1 \\
1 & 0
\end{bmatrix}
\frac{m}{n} + 1
\]

The matrix equation for the CFA is derived in Sec. F, p. 283. We conclude that taking Eq. 2.7 to the \( \left[\frac{m}{n}\right] \)
power, and swapping rows, results in a CFA matrix. However I believe this must be iterated. It follows
that the GCD and CFA are inverses because the matrix formulations are inverses.