MA2K4 Numerical Methods and Computing Assignment 1

Submission Details

This assignment is due to be handed in before

Thursday 1st February 2024, 12 noon

to be submitted via the submission point on the module moodle page.

Student ID	
Student ID of partner	

Please fill in your ID and that of your working partner above. You need to submit your work even when working with a partner.

Please read carefully all instructions on this page to make sure you understand all rules and restrictions. Every submission consists of two parts: A theoretical part, which can be handwritten or computer-written, and a numerical part, which is to be submitted as jupyter notebook `.ipynb`-file. Further rules and restrictions:

- All handwritten parts must be legible to receive full marks.
- Make sure to provide explanations and show your work. The correct answer alone and without explanation will not receive full marks.
- Numerical work should come in an `.ipynb`-file with comments, explanations, code, and figures. The submission file needs to be able to run through from top to bottom to reproduce all the results of your submission.
- Python is the only computer language to be used in this course.
- You are allowed to use pre-defined mathematical functions (such as `exp`, `sin`, ...), but you are NOT allowed to use high-level numerical functions (such as `diff`, `polyfit`, ...).
- All work must be explained, commented on, and all work must be shown. Implementation should contain comments and explanations in markdown surrounding them. Figures should have approprate axis scaling, labels, legends, etc.
- The assignments can be worked on and submitted in pairs. If you choose to do so, indicate your own and your partner's student ID in the fields above. Each partner must upload all submission files to moodle. In this case,both partners receive the same mark.
- Late panelties apply automatically to late submissions. Even if you submit together with a partner, and your partner submits in time, you will be penalised for a late submission if your own submission is not in time.
- To reiterate the above: You will only earn marks if you submit work, and do so before the deadline.

Theoretical Part

This part is to be solved by hand and manual calculation, without the help of a computer. Show all your work to receive full marks.

1.1 For a vector $x \in \mathbb{R}^n$ and matrix $A \in \mathbb{R}^{n\times n}$, and any norm $\|\cdot\|$ on $\mathbb{R}^n$, show that

$$|||A||| = \sup_{x\neq 0} \frac{\|Ax\|}{\|x\|}$$

is a norm on $\mathbb{R}^{n\times n}$.

1.2 Compute the condition number $K(d)$ in the $|\cdot|$-norm on $\mathbb{R}$ for the following equations,

a) $x - a^d = 0$, and

b) $d - x + 1 = 0$,

where d is the datum, a a parameter, and x is the "unknown".

1.3 For the system of equations

$$\begin{cases} x + dy = 1\,, \\ dx + y = 0\,, \end{cases}$$

consider the question: Find the unknowns x and y such that the equation holds for a given datum d.

a) For what values of d is the problem well-posed?

b) Now, consider only x the unknown. What is the condition number $K(d)$ in the $|\cdot|$-norm on $\mathbb{R}$?

c) Instead, consider only y the unknown. What is the condition number $K(d)$ in the $|\cdot|$-norm on $\mathbb{R}$?

1.4 Consider the problem of finding the solution to a quadratic equation,

$$x^2 + 2px - q = 0\,,$$

which has the solutions

$$x_\pm = -p \pm \sqrt{p^2 + q}\,.$$

Study the conditioning of the problem:

a) Consider first p as the datum to compute $K(p)$ in the $|\cdot|$-norm on $\mathbb{R}$. Simplify as much as possible to show that it does not depend on whether one considers x_+ or x_-.

b) Now, consider instead q as the datum to compute $K(q)$ in the $|\cdot|$-norm on $\mathbb{R}$.

1.5 Write down the linear interpolant $p_1(x)$ for the function $f(x) = x^3 - x$, using the nodes $x_0 = 0$ and $x_1 = a$.

Denote by

$$e(x) = f(x) - p_1(x)$$

the error that we produced in the linear interpolation. Show that this error fulfils

$$e(x) = \frac{1}{2} f''(\xi) x(x-a) \, .$$

for the unique ξ that has the value $\xi = \frac{1}{3}(x+a)$.

1.6 Repeat this calculation for the function $f(x) = (x-a)^4$ for the same nodes, and show that in this case there are two possible values for ξ. Give their values.

Numerical Part

This part is to be solved in python in a single jupyter notebook `.ipynb`-file. Make sure that all your explanations, code, figures, results, comments, and discussions are included in this single file to receive all marks. Re-use as much code as possible and avoid any code duplication.

1.7 In this problem, we want to numerically measure the sensitivity of a numerical problem to small perturbations of the data, i.e. numerically measure the phenomenon quantified by the condition number. For this purpose, we want to consider again finding the solutions to quadratic equations of the form

$$x^2 + 2px - q = 0\,,$$

the solutions of which are obviously given by

$$x_\pm = -p \pm \sqrt{p^2 + q}\,.$$

Implement functions `x_plus` and `x_minus` computing the two solutions to the quadratic equation, given p and q. These functions should look like

```python
def x_plus(p, q):
    # insert computation
    return result

def x_minus(p, q):
    # insert computation
    return result
```

which allows you to compute the two results for arbitrary p and q.

a) Consider first, as in problem **1.4 a)**, how sensitive the two solutions are when changing p, to measure the relative change of the answer, $|\delta x|/|x|$ devided by the relative change of the datum p,

$$R_+(p, \delta p) = \frac{|\delta x_+|/|x_+|}{|\delta p|/|p|}\,,$$

and separately the same for x_-. Take a perturbation $\delta p = 10^{-3}$ and compute, for $p = -1$ as well as $p = 1$, for $q = 1$, the quantities R_+ and R_-. These quantities signifies the sensitivity of the answer when changing the input, i.e. p in this case. Comment on your observations and their implication.

b) Now plot $R_+(p, \delta p)$ for $\delta p = 10^{-3}$ and $p \in [-1, 1]$, and the same for R_- in a separate plot. Into each of these figures, also plot the analytical condition numbers $K(p)$ computed in problem **1.4 a)**.

Comment on your observation and comparison. Remember to explain your code and comment on the observed result.

c) Do the same for q: Compute the relative change $|\delta x|/|x|$ divided by the relative change of the datum, $|\delta q|/|q|$, for $p = 1$, $\delta q = 10^{-3}$, for $q = -1$ as well as $q = 1$.

d) Now, for $p = 1$ and $\delta q = 10^{-3}$, plot the numerically measured sensitivity for $q \in [-1, 1]$. Again, plot this for both solutions separately, and compare each against the condition number $K(q)$. Again, comment on your observation and comparison.

1.8 Implement a function `linear_interpolate` that takes a function $f(x)$ and two nodes x_0 and x_1 as well as a position x, and returns its linear interpolant $p_1(x)$ evaluated at x. This function should have the signature

```python
def linear_interpolate(f, x_0, x_1, x):
    # insert computation
    return result
```

This function should allow you to evaluate the linear interpolant of an arbitrary function and with arbitrary nodes at any point x. With this function, do the following:

a) Evaluate the linear interpolant $p_1(x)$ for the nodes $x_0 = 0$, $x_1 = 1$ for the function $f(x) = 2\sin(2x)$ at the location $x = 0.75$.

b) Plot the function $f(x) = \sin(x)$ on the interval $x \in [0,1]$ and its linear interpolant $p_1(x)$ for the nodes $x_0 = 0$, $x_1 = 1$ on the same interval, and interpret your result.

c) Define a new function `interpolation_error` that computes the interpolation error

$$e(x) = f(x) - p_1(x)$$

at a point x. This function has the signature

```python
def interpolation_error(f, x_0, x_1, x):
    return result # compute result first
```

which takes a function f, two nodes x_0, x_1, and a location x at which to compute the error. Make sure to reuse the above function `linear_interpolate` to compute $p_1(x)$ within the function body.

Now, plot this error for the function $f(x) = \ln(x)$ and the nodes $x_0 = 10$, $x_1 = 11$ on the interval $x \in [x_0, x_1]$. What is the error at the specific values $x = 10.2$ and $x = 10.8$?

d) Estimate the maximum absolute error

$$\max_{x\in[x_0,x_1]} |e(x)|$$

of the linear interpolant. Do so for the function $f(x) = 1/(1+x^2)$ with nodes $x_0 = 0$ and $x_1 = h$, by first setting $h = 1$. Approximate the maximum error numerically by computing the error for at least 100 equally spaced gridpoints between x_0 and x_1 to search for its largest absolute value at these points. Implement this procedure in a function `max_error(f, x_0, x_1)`. How large is the maximum absolute error for the above function and nodes?

Now, compute and plot this maximum absolute error for various h given by the formula

$$h = 2^{-k} \quad \text{for} \quad k \in \{0, 1, \ldots, 10\}.$$

Plot the maximum absolute error for each of these h as a function of h on a log-log scale. What do you observe and how do you interpret this result?

Plot for comparison, into the same plot, the function $g(h) = h^\gamma$, and try to find a γ for which the decay rate matches the rate of the absolute error (such that the lines appear mostly parallel). Which γ do you obtain and what does this mean for the *scaling* of the error?

1.9 In this problem, we want to write a functions that allow for the Lagrange interpolation of a generic function.

a) First, write a function

```python
def lagrange(k, x, nodes):
    # code goes here
```

that computes the k-th Lagrange polynomial for the nodes in the array `nodes` and evaluates it at point x, given by the formula

$$L_k(x) = \prod_{j \neq k}^{m} \frac{x - x_j}{x_k - x_j}.$$

Make sure that the function can take a whole array of values in x, for example to facilitate plotting.

Using this function, plot all 5 Lagrange polynomials $L_k(x), k \in \{0, \dots, 4\}$ for the nodes `nodes = [0,1,2,3,4]` on the interval $x \in [0, 4]$. Do the same for all 11 Lagrange polynomials for the nodes `nodes = np.arange(11)` on the interval $x \in [0, 10]$.

b) Using the above function, write a function

```python
def lagrange_interp(x, nodes, values):
    # code  goes here
```

that returns the value of the Lagrange interpolation

$$p_m(x) = \sum_{k=0}^{m} y_k L_k(x)$$

for nodes `nodes` and values `values` and point x. Make sure this function can take an array with many values as x.

Use this to make the following plots: For m equally spaced nodes on the interval $[-5, 5]$, including the boundaries, plot the m-th Lagrange interpolation of $f(x) = 1/(1 + x^2)$ against $x \in [-5, 5]$, and plot $f(x)$ itself for comparison. Do so in four plots for $m = 3, 5, 11, 21$. What do you observe, and what kind of phenomenon is this?

What are the values of each of these interpolants at $x = 1$ and at $x = 4.5$ in comparison to the actual value of the function? Comment on your results. How do your observations relate to the Weierstrass theorem?