# MT3510 Introduction to Mathematical Computing:

# Applied Mathematics Team Project. Sem 2, 2022/23

---

## Polynomial Interpolation

This document outlines the team project on interpolation for MT3510. It is important that everyone reads this document carefully before starting any work.

### Working in your Teams

- Teams are assigned semi-randomly by the module coordinator, please check your email for your Team information. Team allocation will only be changed where clear interpersonal issues prevent you working with a particular individual.
- Each team should nominate a member to be the coordinator who should be responsible for collating the work. Each team coordinator should email or Teams message Dr Lucas to inform them of their nomination.
- The first meeting or interaction of the team should occur, ideally, within 24 hours of the release of the group project. The frequency of meetings or interactions thereafter are at the discretion of each team. Team members should convene on the MS Teams channel to arrange the first meeting, be it virtual or otherwise.
- Each team has an MS Teams channel set up within the module MS Team where you can meet or post discussions and share files.
- Each team member must contribute to the project and complete the peer review form here. *Failure to complete peer review will result in an blanket 0.5 individual multiplier being applied, see below for further instructions.*
- Work on the project should be distributed as evenly as possible between the team members and the allocation should be arrived at as a group in a fair and amicable way (see peer review form). Each section or question should have, at least, one identified group member who is responsible for that part.
- The project "questions" are, by design, less prescriptive than tutorial questions and allow more freedom for you to explore the implementation for yourself. Likewise the questions are, to some extent, open-ended. Some additional commentary or discussion

questions are, to some extent, open-ended. Some additional commentary or discussion about your findings or code is expected, which isn't explicitly asked for in the question.

## Getting help

- Questions and queries are permitted, within reason, and can be directed to Dr Lucas or the week 5 & 6 lab demonstrators. It is suggested that the required instructor is tagged in a post in the team channel when asking questions online so that all team members can view the response. Individual MS Teams chat or email is also allowed.
- Any *serious* interpersonal issues in your team should be reported to the module coordinator immediately.

## Submitting the project

- Submission deadline is 5pm Friday 10th March via mySaint/MMS.
- Submissions should be a single notebook for each team, every team member should submit a copy.
- Make sure your notebook is clearly marked with the team number in the file name and in the header/title text.
- Late submissions will be dealt with using the standard policy (see here).
- The length of the project, when exported, should not exceed around 10 pages, including code, figures and tables.

### Your project report should contain, in markdown:

1. an introduction to the problem (why are we interested in this?).
2. an introduction to the method (how are you solving the problem? How does the method fit in with other methods/problems you know about?).
3. a description of your implementation/code.
4. a description of the results.
5. a discussion/conclusion about what you have found.
6. a short account of "who did what".

Note, not necessarily in this order. For instance it may make sense to have individual discussion/description parts for each section.

## Marking breakdown

| Code 50% | Report 50% |
|---|---|
| gives correct results 25% | description of methods 15% |
| usability & style 10% | discussion of results 15% |

| | |
|---|---|
| efficiency 10% | originality and initiative 10% |
| appropriate outputs 5% | figures/tables/plots 10% |

- This project is worth 20% of the final module grade.
- An individual multiplier will be applied to the team mark based on the peer review scores.
- Projects are open ended and the highest marks are only awarded where independence and initiative has been demonstrated.
- References should be included where appropriate and code should *not* be used verbatim from online sources. Plagiarism checks will be conducted.
- The university *Good Academic Practice* policy applies (see here).

## Peer review instructions

Individual students should distribute a total percentage contribution amongst the team members. For instance if there are $N$ team members and you agree everybody contributed equally, each member should be given $\frac{100}{N}$ of the contribution.

The sum of contributions should be 100% and the partitioning should include yourself.

Justification for any contribution not within $\frac{100}{N} \pm 5$ should be written in the comment box.

You should not mark down a team member where you perceive their contribution was low but where the distribution of work was as agreed upon in the planning stage.

The form is available HERE

---

# Project descrption

### Part 1

- Create a function which performs a *piecewise* Lagrange polynomial interpolation. It should take the polynomial degree, the $(x_k, y_k)$ knots and $x_j$ new evaluation points as input and return the interpolated $y_j$ data, all as `numpy` arrays. Your function should handle unevenly spaced data.
- Validate this function using test knots generated by the function

$$f(x) = e^x \cos(10x)$$

  stored in `x1` and `y1` created via the following code, and choosing appropriate new evaluation points yourself.

In [1]:
```
import numpy as np

def f(x):
    return np.exp(x)*np.cos(10*x)

N = 20
r0 = np.linspace(0,1/6,N)

x1 = 1/(1-r0) # coordinate transformation to test uneven spacing
y1 = f(x1)
```

$N$, $h$, $h$, $h = \frac{b-a}{N}$

`N` or `h` is varied. Note that you may need to tune the choice of evaluation points, or interval, to obtain a robust trend. Your plot should have a different curve for each degree of polynomial tested, and you should attempt to check the rate of convergence from the error data (i.e. the trend $\sim h^n$).

## Part 3

- Create an interactive plot which allows the user to vary the degree of a Lagrange interpolating polynomial (**not** piecewise) for a certain set of knots and evaluation points. The plot should show the interpolating function and the knots (like this). For demonstration purposes use the function from part 1, i.e. $f(x) = e^x \cos(10x)$ but now plot the interval $x \in [1, 2]$, choose a modest number of knots and centre your interpolant (i.e. as the degree increases and more knots are required work from the centre out, as seen in the video).

## Part 4

- Load the data file `wave_data1.txt` (on JupyterHub) using `np.loadtxt`

The data file has a row each for $(t, \xi)$ data where $t$ is time in seconds and $\xi$ is the surface elevation of a water column (in metres) at a particular location in a laboratory wave tank (like this). Unfortunately there are a few data points missing due to a mishap with the wave probe.

- How many data points appear to be missing?
- Use your interpolation function from part 1 to obtain a new uniformly sampled data set on 0.01s subintervals for the first 20s using piecewise cubic polynomials. Plot the new data alongside the equivalent data obtained using `scipy.interpolate.CubicSpline`.
- Plot the difference between the piecewise cubic interpolation and the cubic spline interpolation obtained using `scipy.interpolate.CubicSpline`