AM205 HW1. Data fitting

P1. Polynomial approximation of the gamma function [10 pts]

The gamma function is defined as

$$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t}\, dt$$

and satisfies $(n-1)! = \Gamma(n)$ for integers n.
Note: In Python, the gamma function is available in the `scipy.special` module.

(a) [3 pts] Construct an approximation $g(x)$ to the gamma function as the Lagrange polynomial of the following points:

n	1	2	3	4	6
$\Gamma(n)$	1	1	2	6	120

Write the polynomial as $g(x) = \sum_{k=0}^{4} g_k x^k$. Report the values of the coefficients g_k.

(b) [3 pts] Construct another approximation $h(x)$ to the gamma function by first calculating the fourth order polynomial $p(x)$ that interpolates the points $(n, \log(\Gamma(n)))$ for $n = 1, 2, 3, 4, 6$. Then define the approximation as $h(x) = \exp(p(x))$. Report the values of the coefficients of $p(x)$.

(c) [2 pts] Plot values of $\Gamma(x)$, $g(x)$, and $h(x)$ and the relative errors $|\Gamma(x) - g(x)|/\Gamma(x)$ and $|\Gamma(x) - h(x)|/\Gamma(x)$ on the interval $1 \le x \le 6$. Use a logarithmic scale for the error.

(d) [2 pts] Calculate the maximum relative error between $\Gamma(x)$ and $g(x)$ on the interval $1 \le x \le 6$ by evaluating the functions on a uniform grid with 1001 points. Repeat this for $\Gamma(x)$ and $h(x)$. Which of the two approximations is more accurate?

P2. Error bounds with Lagrange polynomials [15 pts]

(a) [3 pts] Let $f(x) = e^{-4x} + e^{3x}$. Write a program to calculate and plot the Lagrange polynomial $p_{n-1}(x)$ of $f(x)$ at the Chebyshev points $x_j = \cos((2j-1)\pi/2n)$ for $j = 1, \ldots, n$. For $n = 4$, over the range $[-1, 1]$, plot $f(x)$ and Lagrange polynomial $p_3(x)$.

(b) [4 pts] Recall from the lectures that the infinity norm for a function g on $[-1, 1]$ is defined as $\|g\|_\infty = \max_{x\in[-1,1]} |g(x)|$. Calculate $\|f - p_3\|_\infty$ by sampling the function at 1001 equally-spaced points over $[-1, 1]$.

(c) [6 pts] Recall the interpolation error formula from the lectures,

$$f(x) - p_{n-1}(x) = \frac{f^{(n)}(\theta)}{n!}(x - x_1)(x - x_2)\ldots(x - x_n)$$

for some $\theta \in [-1, 1]$. Use this formula to derive an upper bound for $\|f - p_{n-1}\|_\infty$ for any positive integer n. Your bound should be a mathematical formula, and should not rely on numerical sampling.

(d) [2 pts] Find a cubic polynomial $p_3^\dagger$ such that $\|f - p_3^\dagger\|_\infty < \|f - p_3\|_\infty$.

P3. Condition number of a matrix [8 pts]

For a 2×2 invertible matrix A, define the condition number to be $\kappa(A) = \|A\| \, \|A^{-1}\|$ as discussed in the lectures. Assume that the matrix norm is defined using the Euclidean vector norm.

(a) [2 pts] Find two 2×2 invertible matrices B and C such that $\kappa(B + C) < \kappa(B) + \kappa(C)$.

(b) [2 pts] Find two 2×2 invertible matrices B and C such that $\kappa(B + C) > \kappa(B) + \kappa(C)$.

(c) [4 pts] Suppose that Q is an orthogonal matrix, i.e. $Q^\mathsf{T} = Q^{-1}$, and $\alpha \in \mathbb{R}$. Find $\kappa(QA)$ in terms of $\kappa(A)$ and Q. Find $\kappa(\alpha A)$ in terms of $\kappa(A)$ and α.

P4. Periodic cubic splines [15 pts]

In the lectures we discussed the construction of cubic splines to interpolate between a number of control points. We found that it was necessary to impose additional constraints at the end points of the spline in order to have enough constraints to determine the cubic spline uniquely. Here, we examine the construction of cubic splines on a periodic interval $t \in [0, 4)$, where $t = 0$ is equivalent to $t = 4$. Working in a periodic interval simplifies the spline construction and requires no additional constraints.
Note: Do not use existing modules, such as `scipy.interpolate`, for constructing the spline. However, you may use other modules for solving linear systems and numerical integration (e.g. `numpy.linalg.solve` and `scipy.integrate`).

(a) [5 pts] Consider four points $(t, x) = (0, 0), (1, 4), (2, 0), (3, -4)$. Construct a cubic spline $s_x(t)$ that is piecewise cubic in the four intervals $[0, 1)$, $[1, 2)$, $[2, 3)$, and $[3, 4)$. At $t = 0, 1, 2, 3$ the cubics should match the control points, giving eight constraints. At $t = 0, 1, 2, 3$ the first and second derivatives should match, giving and additional eight constraints and allowing $s_x(t)$ to be uniquely determined.

(b) [1 pts] Plot $s_x(t)$ and $4\sin(t\pi/2)$ on the interval $[0, 4)$ and show that they are similar.

(c) [6 pts] Construct a second cubic spline $s_y(t)$ that goes through the four points $(t, y) = (0, 2), (1, 0), (2, -2), (3, 0)$. Plot $s_y(t)$ and $2\cos(t\pi/2)$ on the interval $0 \le t < 4$ to see how similar they are.

(d) [3 pts] In the xy-plane, plot the parametric curve $(s_x(t), s_y(t))$ for $t \in [0, 4)$. Calculate the area enclosed by the parametric curve to at least five decimal places. Use the area to estimate π from the relationship $A = r_1 r_2 \pi$ where $r_1 = 2$ and $r_2 = 4$.

P5. Image reconstruction from low light [24 pts]

In the archive `am205_hw1_data.zip`, you will find a directory called `p5_fragments` that contains several photo fragments of a scene with different objects taken from a bird feeder station in the Netherlands. The filename of each image (e.g. `0258_frag0.png`) consists of two parts. The first part is a timestamp (hours and minutes). The second part denotes one of the four fragments showing different objects. Images `0258`, `0646`, and `0704` were taken in low light, while image `0927` was taken in regular light.

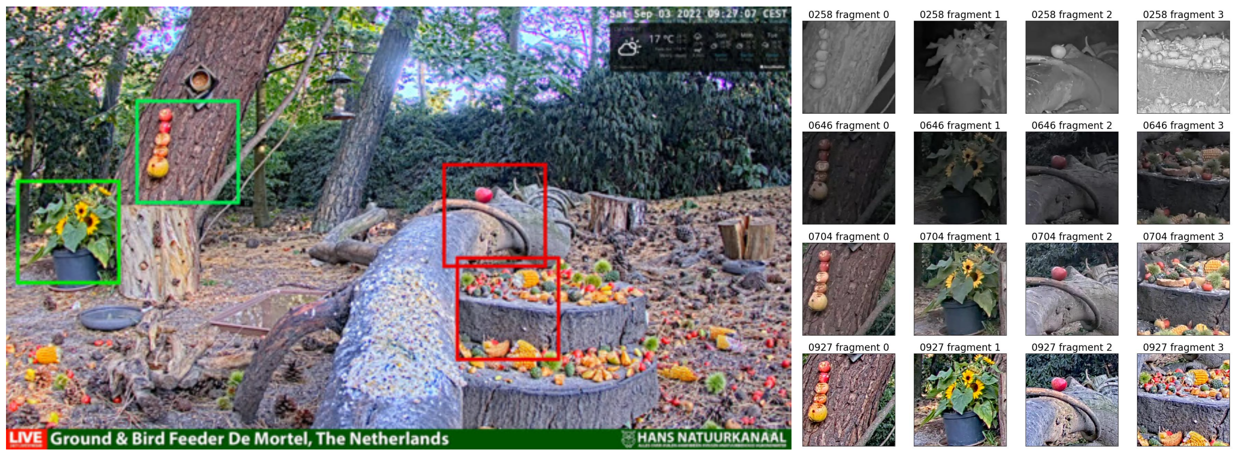

Figure 1: Full scene at time `0927` (left) and all extracted fragments (right).

The size of each image is 256×256 pixels. Each pixel in the image can be represented as a vector $\mathbf{p} \in \mathbb{R}^3$ containing three intensity values between 0 and 1 for the red, green, and blue components. Assume that all pixels in fragments having the same timestamp are indexed and K_{all} is the set of all possible indices, so that $|K_{\text{all}}| = 4 \cdot 256^2$. Let K_0, K_1, K_2, $K_3 \subset K_{\text{all}}$ denote the subsets of indices corresponding to fragments 0, 1, 2, and 3 respectively, so that $|K_0| = |K_1| = |K_2| = |K_3| = 256^2$. Then, let $\mathbf{p}_k^A$, $\mathbf{p}_k^B$, $\mathbf{p}_k^C$, and $\mathbf{p}_k^D$ be the k-th pixel of images with timestamps `0258`, `0646`, `0704`, and `0927` respectively.

(a) [13 pts] Consider reconstructing the regular-light photo $\mathbf{p}^D$ from the three low-light photos $\mathbf{p}^A$, $\mathbf{p}^B$, and $\mathbf{p}^C$ using the following model

$$\mathbf{p}_k = F^A \mathbf{p}_k^A + F^B \mathbf{p}_k^B + F^C \mathbf{p}_k^C + \mathbf{p}_{\text{const}}, \tag{1}$$

where F^A, F^B, and F^C are 3×3 matrices and $\mathbf{p}_{\text{const}} \in \mathbb{R}^3$. Find a least-squares fit for the unknown F^A, F^B, F^C, and $\mathbf{p}_{\text{const}}$ using pixels from fragments 0 and 1. Specifically, you should minimize the error

$$S_{ABC}(K) = \frac{1}{|K|} \sum_{k \in K} \| F^A \mathbf{p}_k^A + F^B \mathbf{p}_k^B + F^C \mathbf{p}_k^C + \mathbf{p}_{\text{const}} - \mathbf{p}_k^D \|_2^2$$

for $K = K_0 \cup K_1$. Report the obtained values of F^A, F^B, F^C, and $\mathbf{p}_{\text{const}}$. Visualize the output of the fitted model (1) for all four fragments together with the reference images `0927`. Some pixel intensities you obtain from (1) may lie outside the range between 0 and 1, in which case you need to clip them, e.g. using `numpy.clip`. Calculate the error S_{ABC} for each fragment, i.e. report values of $S_{ABC}(K_0)$, $S_{ABC}(K_1)$, $S_{ABC}(K_2)$, and $S_{ABC}(K_3)$.

(b) [8 pts] Repeat part **(a)** using only images `0646` as input. In this case, the model simplifies to

$$\mathbf{p}_k = F^B \mathbf{p}_k^B + \mathbf{p}_{\text{const}}, \tag{2}$$

and the fitting error takes the form

$$S_B(K) = \frac{1}{|K|} \sum_{k \in K} \| F^B \mathbf{p}_k^B + \mathbf{p}_{\text{const}} - \mathbf{p}_k^D \|_2^2.$$

(c) [3 pts] Compare results from parts **(a)** and **(b)**. Does using multiple light levels as input improve the quality of the fit compared to using a single light level?

P6. Determining hidden chemical sources [20 pts]

Suppose that $\rho(\mathbf{x}, t)$ represents the concentration of a chemical diffusing in two-dimensional space, where $\mathbf{x} = (x, y)$. The concentration satisfies the diffusion equation

$$\frac{\partial \rho}{\partial t} = \frac{\partial^2 \rho}{\partial x^2} + \frac{\partial^2 \rho}{\partial y^2}. \tag{3}$$

If a localized point source of chemical is introduced at the origin at $t = 0$, its concentration satisfies

$$\rho_c(\mathbf{x}, t) = \frac{1}{4\pi t} \exp\left(-\frac{x^2 + y^2}{4t} \right).$$

(a) [3 pts] Show by direct calculation that the concentration ρ_c satisfies (3).

(b) [7 pts] Suppose that 49 point sources of chemicals are introduced at $t = 0$ with different strengths, on a 7×7 regular lattice covering the coordinates $x = -3, -2, \ldots, 3$ and $y = -3, -2, \ldots, 3$. By linearity of (3) the concentration will satisfy

$$\rho(\mathbf{x}, t) = \sum_{k=0}^{48} \lambda_k \rho_c(\mathbf{x} - \mathbf{v}_k, t),$$

where $\mathbf{v}_k$ is the k-th lattice site and λ_k is the strength of the chemical introduced at that site. In the archive `am205_hw1_data.zip`, you will find a file `p6_data.txt` with many measurements of $\rho(\mathbf{x}, t)$ at $t = 4$. The file contains three columns: position x_i, position y_i, and concentration $\rho(\mathbf{x}_i, 4)$ for $i = 0, \ldots, 199$. By any means necessary, determine the concentration strengths λ_k.

(c) [6 pts] Suppose that the measurements have some experimental error, so that the measured values $\tilde{\rho}_i$ in the file are related to the true values ρ_i according to

$$\tilde{\rho}_i = \rho_i + e_i$$

where the e_i are normally distributed with mean 0 and variance 10^{-8}. Construct a hypothetical sample of the true ρ_i, and repeat your procedure from part **(b)** to determine the concentrations λ_k. Repeat this sampling procedure for at least 100 times, and use it to measure the standard deviation in the λ_k at the lattice sites $(0, 0)$, $(1, 1)$, $(2, 2)$, $(3, 3)$. Which of these has the largest standard deviation and why?

(d) [4 pts] You should find that the concentrations λ_k from part **(b)** take values in the range between 0 and 32. Round each λ_k to the nearest integer and convert the integer to its binary representation consisting of five bits. The bits encode monochrome images. To recover the first image, take the most significant (fifth bit, even if zero) bit from the 5-bit binary representation of all λ_k and draw them on a 7*x*7 grid. Repeating this for all bits will give you five images. What message can you read from them?