PURDUE UNIVERSITY · CS 31400
NUMERICAL ANALYSIS

HOMEWORK

David F. Gleich
February 11, 2021

Please answer the following questions in complete sentences in a typed manuscript and submit the solution on gradescope by on February 22nd at around 5am

Problem 0: Homework checklist

- Please identify anyone, whether or not they are in the class, with whom you discussed your homework. This problem is worth 1 point, but on a multiplicative scale.
- Make sure you have included your source-code and prepared your solution according to the most recent Piazza note on homework submissions.

Problem 1: Gautschi Exercise 2.2

Consider the data

$$f(t_i) = 1, i = 1, \ldots, N-1, f(t_N) = y \gg 1.$$

1. Determine the discrete L_∞ approximant to f by means of a constant c.
2. Do the same for discrete (equally weighted) least squares approximation.
3. Compare what happens as $N \to \infty$.

Problem 2: Gautschi Exercise 2.5

Taylor expansion yields the simple approximation $e^x \approx 1 + x$ on $0 \le x \le 1$. Suppose you want to improve this by seeking an approximation of the form $e^x \approx 1 + cx$ for $0 \le x \le 1$ and some suitable c.

1. How must c be chosen if the approximation is to be optimal in the continuous, equally weighted least-squares sense?
2. Plot the error curves $e_1(x) = e^x - (1 + x)$ and $e_2(x) = e^x - (1 + cx)$ using c determined in part 1. Find the maximum magnitude of each error curve on $0 \le x \le 1$.
3. Repeat steps 1 and 2 for polynomial approximations $e^x \approx 1 + cx + dx^2$.

Problem 3: Gautschi Exercise 2.11

Suppose you want to approximate the step function

$$f(t) = \begin{cases} 1 & 0 \le t \le 1 \\ 0 & t > 1 \end{cases}$$

on the positive real line $\mathbb{R}_+$ using a linear combination of exponentials $\pi_j(t) = e^{-jt}, j = 1, \ldots, n$ in a least-squares sense.

1. Derive the normal equations. How is the matrix related to the Hilbert matrix?
2. Plot the condition number of the matrix for $1 \le n \le 4$.

Problem 4: Gautschi Machine Exercise 2.2

1. Determine the $(n+1)\times(n+1)$ matrix $\boldsymbol{A} = [a_{ij}], a_{ij} = (B_j^n, B_i^n)$ of the normal equations relative to the Berstein basis

$$B_j^n(t) = \binom{n}{j} t^j (1-t)^{n-j} \qquad j = 0, \ldots, n,$$

   and weight function $w(t) = 1$ on $[0,1]$.

2. Solve the normal equations for $n = 5:5:25$ on a computer when $f(t) = 1$. What should the exact answer be? For each n, print the infinity norm of the error vector and an estimate of the condition number of $\boldsymbol{A}$.

Problem 5: Chebyshev approximation

There are two types of Chebyshev points commonly used. Your book discusses both, although doesn't call them by their names "Chebyshev points of the first kind" and "Chebyshev points of the second kind". The Chebyshev points of the first kind are the roots, or zeros, of the nth Chebyshev polynomial (equation 2.85). Chebyshev points of the second kind are the extrema of the nth Chebyshev polynomial. Note that an n degree polynomial will have $n+1$ extrema.

1. Develop a program to compute the Chebyshev points of the first and second kind. Prepare a plot of each set of points for $n = 11, 101, 1001, 10001$. Discuss any features of your plot.

2. Generate a program to compute a polynomial interpolant through Chebyshev points of the second kind and uniformly spaced points on an arbitrary interval $[a,b]$ and evaluate the resulting interpolant at another set of points.

   Your program should take as input a "function handle", a function handle, the number of points to use to build the interpolant, and the set of points.

```
"""
Compute an interpolant for function f in n uniformly spaced
points over the region [r(1),r(2)] and return the values
of the polynomial at points pts.

Example:
  f(x) = e^(-x) # declare a function in julia to evaluate e^-x
  pts = linspace(1,5,10000) # generate points to evaluate the polynomial
  y = interpolate_unif(f, [1,5], 101, pts) # build the interpolant and evaluate
  plot(pts, y)
"""

function interpolate_unif(f, r, n, pts)
    # you have to fill this in
end
```

3. Apply your program to the function $f(x) = e^{-x}$ on the region $[1,5]$ with $n = 5, 8, 11, 30, 51$. Prepare plots of your results and discuss them.

4. Apply your program to the function $f(x) = \frac{1}{x^2+5}$ on the region $[-5,5]$ with $n = 5, 8, 11, 30, 51$. Prepare plots of your results and discuss them.

5. Discuss the course logo on the course homepage in the context of your results.

Problem 6: Multivariate approximation

The book has largely focused on univariate approximation where we have a univariate function $f(x)$ and we wish to generate a computer approximation of that function. In many scientific studies, we have multivariate data where f may depend on multiple inputs say, x and y. The same idea of having a linear function space works, but we need to enrich it with multivariate functions!

The degree of a multivariate polynomial is the sum of powers of expressions. Suppose we work with the space of bi-variate (two variable) polynomials of degree at most 1. These are: $\{f(x,y) = c_0 + c_1 x + c_2 y \mid c_i \in \mathbb{R}\}$ If we increase it degree 2, we get $\{f(x,y) = c_0 + c_1 x + c_2 y + c_3 xy + c_4 x^2 + c_5 y^2 \mid c_i \in \mathbb{R}\}$

We extend the idea of a function norm in the same way:

$$\|f\|_2 = \left(\int_{x_a}^{x_b} \int_{y_a}^{y_b} f(x,y)^2 \, dxdy \right)^{1/2}.$$

Thus, the best approximation problem easily translates into the multivariate case, and these generalize in a straightforward way at this point.

1. (Warmup) Consider approximating the bivariate step function $f(x,y) = \begin{cases} 1 & x,y > 0 \\ -1 & x,y < 0 \\ 0 & \text{otherwise} \end{cases}$ with a constant in the L_2 sense. Show that the best approximation is the constant 0.

2. How many points uniquely determine the interpolating polynomial of degree at most d if there are two variables?

3. How many points uniquely determine the interpolating polynomial of degree at most d if there are k variables? (no need for a closed form expression, but your answer should be recogonizably correct and well explained)

4. Derive the Lagrange form of the interpolating polynomial in the bivariate case, e.g.

$$p(x,y) = \sum_{i=0}^{n} f_i \ell_i(x,y).$$

5. One way to sample a function in multiple dimensions is to use a tensor-product construction. We'll illustrate a simple case. Suppose we have the univariate points $[0.2, 0.5, 0.7]$, then the tensor-product point (sometimes called Cartesian product) is the set

$$(x,y) = (0.2,0.2),(0.2,0.5),(0.2,0.7),(0.5,0.2),(0.5,0.5),(0.5,0.7),(0.7,0.2),(0.7,0.5),(0.7,0.7).$$

Use a tensor product of 5 Chebyshev points on the region $[-5,5]$ to sample the Multivariate Runge function :

$$f(x,y) = \frac{1}{x^2 + y^2 + 5}$$

and display your interpolant. (Hint, see here http://nbviewer.jupyter.org/github/gizmaa/Julia_Examples/blob/master/pyplot_surfaceplot.ipynb for info on plotting.)