

MATH 4640 Numerical Analysis - HW 3 Solutions

Austin Barton

March 27, 2025

Problem 1: Please see attached PDF with all written solutions for questions 1-4 and 6-7.

Problem 2: Please see attached PDF with all written solutions for questions 1-4 and 6-7.

Problem 3: Please see attached PDF with all written solutions for questions 1-4 and 6-7.

Problem 4: Please see attached PDF with all written solutions for questions 1-4 and 6-7.

Problem 5:

```
43 fn func1(x: f64) -> f64 {
44     (5.0 * x).sin().powi(2)
45 }
46
47 fn func2(x: f64) -> f64 {
48     (3.0 * x).cos().powi(3) + (SQRT_2 * x).cos()
49 }
50
51 fn reconstruct(function:fourier_coefficients: &[Complex<f64>], n: usize, x: f64, m: i32) -> f64 {
52     let mut sum = 0.0;
53     for j in 0..m {
54         let idx = if j < 0 {
55             ((-j + n as i32) as usize) % n
56         } else {
57             j as usize % n
58         };
59         let coef = fourier_coefficients[idx];
60         sum += coef.re * (j as f64 * x).cos() - coef.im * (j as f64 * x).sin();
61     }
62     sum
63 }
64
65 fn calculate_and_print_fourier_coefficients=>F(>
66     func: F,
67     n: usize,
68     description: &str,
69     theoretical_values: Option<&[(&str, f64)]>,
70     filename: &str,
71     where
72         F: Fn(f64) -> f64,
73         &str: std::borrow::Borrow<&str>
74     {
75         let x_k = [k: usize] -> f64 { 2.0 * PI * k as f64 / n as f64 };
76
77         let mut buffer: VecComplex<f64> = (0..n)
78             .map(|k| Complex {
79                 re: func(x_k(k)),
80                 im: 0.0,
81             })
82             .collect();
83
84         let mut planner = FftPlanner::<f64>::new();
85         let fft = planner.plan_fft_forward(n);
86         fft.process(&mut buffer);
87
88         buffer.iter_mut().for_each(|coef| *coef /= n as f64);
89
90         println!("Fourier coefficients for {}", description);
91         println!("");
92
93         for j in -100..=100 {
94             let idx = if j < 0 {
95                 ((-j + n as i32) as usize) % n // Ensure proper wrapping
96             } else {
97                 j as usize % n
98             };
99             let coef = &function(x_k(j));
100            let value = buffer[idx].re + buffer[idx].im * I;
101            if let Some((expected, _)) = theoretical_values {
102                assert!(approx_eq!(f64, value, expected, epsilon));
103            }
104        }
105    }
106 }
```

Figure 1: Screenshot of code for calculating fourier coefficients using FFT.

```

10 |     let mut planner = FftPlanner::new();
11 |     let fft = planner.plan_fft_forward();
12 |     fft.process(&mut buffer);
13 |
14 |     buffer.iter_mut().for_each(|coef| *coef /= n as f64);
15 |     println!("Fourier coefficients for {}", description);
16 |     println!("-----");
17 |
18 |     for j in -100..=100 {
19 |         let idx = if j < 0 {
20 |             ((j + n as i32) as usize) % n // Ensure proper wrapping
21 |         } else {
22 |             j as usize % n
23 |         };
24 |
25 |         let threshold = 1e-12;
26 |         if buffer[idx].re.abs() > threshold || buffer[idx].im.abs() > threshold {
27 |             println!("{}={:.12}+{:i.12}j", j, buffer[idx].re, buffer[idx].im);
28 |         }
29 |     }
30 |
31 |     plot_functions(func, &buffer, filename); // use 'let _ = ...' to ignore the resulting value: 'let _ = '
32 |
33 |     if let Some(values) = theoretical_values {
34 |         println!("---Theoretical Values---");
35 |         for (label, value) in values {
36 |             println!("{} = {:.12}, label, value");
37 |         }
38 |         println!("All other coefficients = 0");
39 |     }
40 |
41 |     println!("-----");
42 |
43 | fn solve_q5() -> Result<(), Box<dyn std::error::Error>> {
44 |     let n = 200;
45 |
46 |     let func1_theory = [
47 |         ("f_hat(0)", 0.5),
48 |         ("f_hat(10)", -0.25),
49 |         ("f_hat(-10)", -0.25),
50 |     ];
51 |
52 |     calculate_and_print_fourier_coefficients(
53 |         func1,
54 |         n,
55 |         "f(x) = sin^2(5x)",
56 |         Some(&func1_theory),
57 |         "plots/first-func-plot.png",
58 |     );
59 |
60 |     println!("-----");
61 |
62 |     plot_functions(func, &buffer, filename); // use 'let _ = ...' to ignore the resulting value: 'let _ = '
63 |
64 |     if let Some(values) = theoretical_values {
65 |         println!("---Theoretical Values---");
66 |         for (label, value) in values {
67 |             println!("{} = {:.12}, label, value");
68 |         }
69 |         println!("All other coefficients = 0");
70 |     }
71 |
72 |     println!("-----");
73 |
74 | fn solve_q5() -> Result<(), Box<dyn std::error::Error>> {
75 |     let n = 200;
76 |
77 |     let func1_theory = [
78 |         ("f_hat(0)", 0.5),
79 |         ("f_hat(10)", -0.25),
80 |         ("f_hat(-10)", -0.25),
81 |     ];
82 |
83 |     calculate_and_print_fourier_coefficients(
84 |         func1,
85 |         n,
86 |         "f(x) = sin^2(5x)",
87 |         Some(&func1_theory),
88 |         "plots/first-func-plot.png",
89 |     );
90 |
91 |     calculate_and_print_fourier_coefficients(
92 |         func2,
93 |         n,
94 |         "f(x) = cos^2(3x) + cos(\sqrt{2}x)",
95 |         None,
96 |         "plots/second-func-plot.png",
97 |     );
98 |
99 |     Ok(())
100 |
101 |
102 | pub fn main() {
103 |     if let Err(e) = solve_q5() {
104 |         eprintln!("Error: {}", e);
105 |         std::process::exit(1);
106 |     }
107 | }

```

Normal ↵ hw3 ↵ □ WI HI ↵ -/Documents/Academics/Math/Numerical/numerical_analysis_math4640/solutions/rust/hw3/src/q5.rs
numerical@nvim: 1:nvim 2:nvim 3:nvim*

rust utf-8[unix] 5.09KiB 102:1
"cabin" 22:32 27-Mar-29

Figure 2: Screenshot of code for calculating fourier coefficients using FFT.

```

10 |
11 |
12 |
13 |
14 |
15 |
16 |
17 |
18 |
19 |
20 |
21 |
22 |
23 |
24 |
25 |
26 |
27 |
28 |
29 |
30 |
31 |
32 |
33 |
34 |
35 |
36 |
37 |
38 |

```

Normal ↵ hw3 ↵ □ WI HI ↵ -/Documents/Academics/Math/Numerical/numerical_analysis_math4640/solutions/rust/hw3/src/q5.rs
numerical@nvim: 1:nvim 2:nvim 3:nvim*

rust utf-8[unix] 5.09KiB 102:1
"cabin" 22:32 27-Mar-29

Figure 3: Screenshot of code for calculating fourier coefficients using FFT.

```

warning: 'hw0' (bin "hw0") generated 4 warnings
  Finished 'dev' profile [unoptimized + debuginfo] target(s) in 1.73s
    Running target/debug/hw0 q3
---MATH 4640, Numerical Analysis, Homework 3---

-----
Running solution to question 5...
Fourier coefficients for f(x) = sin^2(5x)

f_hat(10): -0.250000000000 + 1.0e-000000000000i
f_hat( 0): 0.500000000000 + 1.0e-000000000000i
f_hat(-10): -0.250000000000 + 1.0e-000000000000i
Plot saved to plots/first-func-plot.png

--Theoretical Values--
f_hat(0) = 0.500000000000
f_hat(10) = -0.250000000000
f_hat(-10) = -0.250000000000
All other coefficients = 0

-----
Fourier coefficients for f(x) = cos^2(3x) + cos(\sqrt{2}x)

f_hat(-100): 0.00461792771 + 1.0e-000000000000i
f_hat(-99): 0.004617622732 + 1.0e-000073014015i
f_hat(-98): 0.0046176016008 + 1.0e-000146641380i
f_hat(-97): 0.004616966333 + 1.0e-000219186514i
f_hat(-96): 0.004616653889 + 1.0e-000295174820i
f_hat(-95): 0.004616552089 + 1.0e-000375535054i
f_hat(-94): 0.004616774886 + 1.0e-000439351577i
f_hat(-93): 0.004616682998 + 1.0e-000513128705i
f_hat(-92): 0.00461674537 + 1.0e-000587162568i
f_hat(-91): 0.004616729998 + 1.0e-000662303300i
f_hat(-90): 0.0046165114208 + 1.0e-000736153655i
f_hat(-89): 0.004616160893 + 1.0e-000811189093i
f_hat(-88): 0.00461591752 + 1.0e-000886637624i
f_hat(-87): 0.004615700000 + 1.0e-000961557100i
f_hat(-86): 0.004615694527 + 1.0e-001038939545i
f_hat(-85): 0.004615385619 + 1.0e-001115877897i
f_hat(-84): 0.004615149248 + 1.0e-001193399701i
f_hat(-83): 0.004614905594 + 1.0e-001271350594i
f_hat(-82): 0.004614622913 + 1.0e-00135036703i
f_hat(-81): 0.004614329998 + 1.0e-001429927154i
f_hat(-80): 0.004614018174 + 1.0e-001510251421i
f_hat(-79): 0.004613604956 + 1.0e-001591401242i
f_hat(-78): 0.004613295088 + 1.0e-001671301609i
f_hat(-77): 0.004612956586 + 1.0e-001756393802i
f_hat(-76): 0.004612575911 + 1.0e-001840349954i
f_hat(-75): 0.004612135280 + 1.0e-001925358824i
f_hat(-74): 0.004611808082 + 1.0e-002010831808i
f_hat(-73): 0.004611213902 + 1.0e-002098788602i
f_hat(-72): 0.004610712828 + 1.0e-002187343618i
f_hat(-71): 0.004610183100 + 1.0e-002277220024i
f_hat(-70): 0.004609623330 + 1.0e-002366710000i
f_hat(-69): 0.004609141376 + 1.0e-0024555541376i
f_hat(-68): 0.004608407439 + 1.0e-002555549472i
[numerical0:nvim- 1:nvim- 2:nvim- 3:[tmux]*]

```

Figure 4: Screenshot of result of code for calculating fourier coefficients using FFT.

```

Fourier coefficients for f(x) = cos^2(3x) + cos(\sqrt{2}x)

f_hat(-100): 0.00461792771 + 1.0e-000000000000i
f_hat(-99): 0.004617622732 + 1.0e-000073014015i
f_hat(-98): 0.0046176016008 + 1.0e-000146641380i
f_hat(-97): 0.004616966333 + 1.0e-000219186514i
f_hat(-96): 0.004616653889 + 1.0e-000295174820i
f_hat(-95): 0.004616552089 + 1.0e-000375535054i
f_hat(-94): 0.004616774886 + 1.0e-000439351577i
f_hat(-93): 0.004616682998 + 1.0e-000513128705i
f_hat(-92): 0.00461674537 + 1.0e-000587162568i
f_hat(-91): 0.004616729998 + 1.0e-000662303300i
f_hat(-90): 0.0046165114208 + 1.0e-000736153655i
f_hat(-89): 0.004616160893 + 1.0e-000811189093i
f_hat(-88): 0.004614905594 + 1.0e-000886637624i
f_hat(-87): 0.004614622913 + 1.0e-000961557100i
f_hat(-86): 0.004614329998 + 1.0e-001038939545i
f_hat(-85): 0.004614018174 + 1.0e-001115877897i
f_hat(-84): 0.004613604956 + 1.0e-001193399701i
f_hat(-83): 0.004613295088 + 1.0e-001271350594i
f_hat(-82): 0.004612575911 + 1.0e-001350367032i
f_hat(-81): 0.004612135280 + 1.0e-001429927154i
f_hat(-80): 0.004611808082 + 1.0e-001510251421i
f_hat(-79): 0.004611213902 + 1.0e-001591401242i
f_hat(-78): 0.004611332278 + 1.0e-001671301609i
f_hat(-77): 0.004612956586 + 1.0e-001756393802i
f_hat(-76): 0.004612575911 + 1.0e-001840349954i
f_hat(-75): 0.004612135280 + 1.0e-001925358824i
f_hat(-74): 0.004611808082 + 1.0e-002010831808i
f_hat(-73): 0.004611213902 + 1.0e-002098788602i
f_hat(-72): 0.004610712828 + 1.0e-002187343618i
f_hat(-71): 0.004610183100 + 1.0e-002277220024i
f_hat(-70): 0.004609623330 + 1.0e-002366710000i
f_hat(-69): 0.004609141376 + 1.0e-0024555541376i
f_hat(-68): 0.004608407439 + 1.0e-002555549472i
[numerical0:nvim- 1:nvim- 2:nvim- 3:[tmux]*]

```

Figure 5: Screenshot of result of code for calculating fourier coefficients using FFT.

Thursday March 27, 22:45

ta numerical

```
f_hat(-28) : 0.004487921679 + 10.009899218028
f_hat(-27) : 0.004487672248 + 10.0100187963772
f_hat(-26) : 0.004487419500 + 1.0107900000000000
f_hat(-25) : 0.0044850295522 + 10.012332409330
f_hat(-24) : 0.0044843489187 + 10.011776171717
f_hat(-23) : 0.0044840161993 + 10.012342807198
f_hat(-22) : 0.0044838512950 + 10.012342807193
f_hat(-21) : 0.004372600323 + 10.013632620895
f_hat(-20) : 0.00434569991 + 10.014371716465
f_hat(-19) : 0.0043414654594 + 10.015187234163
f_hat(-18) : 0.0043373155860 + 10.015986659989
f_hat(-17) : 0.0043233155846 + 10.017102425598
f_hat(-16) : 0.004181072469 + 10.018238689786
f_hat(-15) : 0.004117857793 + 10.019526859658
f_hat(-14) : 0.00404040427383 + 10.021090698567
f_hat(-13) : 0.004000000000000000 + 10.021090698567
f_hat(-12) : 0.003822375307 + 10.024699864522
f_hat(-11) : 0.003665136767 + 10.027069582943
f_hat(-10) : 0.003457108622 + 10.029934323723
f_hat(-9) : 0.003257108622 + 10.032809333333
f_hat(-8) : 0.002772613226 + 10.037965737380
f_hat(-7) : 0.002177919813 + 10.043876599887
f_hat(-6) : 0.001238065958 + 10.052044124148
f_hat(-5) : 0.00093131072469 + 10.061702425598
f_hat(-4) : 0.008316144640 + 10.084040377333
f_hat(-3) : 0.363131079683 + 10.126674563415
f_hat(-2) : -0.053129203583 + 10.295695635292
f_hat(-1) : -0.062401173173 + 10.29576861887
f_hat(0) : 0.002401173173 + 10.29576861887
f_hat(1) : 0.120166517395 + 10.295768618877
f_hat(2) : -0.053129203583 + 10.295695635292
f_hat(3) : 0.363131079683 + 10.126674563415
f_hat(4) : 0.00093131072469 + 10.061702425598
f_hat(5) : 0.00080387037921 + 10.064170553199
f_hat(6) : 0.001238065958 + 10.052044124148
f_hat(7) : 0.002177919813 + 10.043876599887
f_hat(8) : 0.003257108622 + 10.032809333333
f_hat(9) : 0.002772613226 + 10.037965737380
f_hat(10) : 0.003457108622 + 10.029934323723
f_hat(11) : 0.003665136767 + 10.027069582943
f_hat(12) : 0.003257108622 + 10.032809333333
f_hat(13) : 0.002772613226 + 10.037965737380
f_hat(14) : 0.00404040427383 + 10.021090698567
f_hat(15) : 0.004117857793 + 10.019526859658
f_hat(16) : 0.004181072469 + 10.018238689786
f_hat(17) : 0.004277893941 + 10.01692061998
f_hat(18) : 0.00437108622 + 10.027069582943
f_hat(19) : 0.004487921679 + 10.011776171717
f_hat(20) : 0.0044843489187 + 10.014371716465
f_hat(21) : 0.0044838512950 + 10.015187234163
f_hat(22) : 0.004372600323 + 10.01692061998
f_hat(23) : 0.0044651651993 + 10.012342807198
f_hat(24) : 0.0044843489187 + 10.011776171717
f_hat(25) : 0.0044838512950 + 10.0107900000000000
f_hat(26) : 0.00448401173173 + 10.0107900000000000
f_hat(27) : 0.004487672248 + 10.010181796372
numerical@nvin:~$ vim 2:nvim 2:nvim 3:[tmux]*
```

"cabin" 22:45 27-Mar-25

Figure 6: Screenshot of result of code for calculating fourier coefficients using FFT.

Thursday March 27, 22:45

ta numerical

```
f_hat( 52) : 0.004591864581 + 1.0.004366507159
f_hat( 53) : 0.004593402389 + 1.0.004239932542
f_hat( 54) : 0.004593402389 + 1.0.004239932542
f_hat( 55) : 0.004593402389 + 1.0.004239932542
f_hat( 56) : 0.004593402389 + 1.0.004239932542
f_hat( 57) : 0.004593701425 + 1.0.003724797114
f_hat( 58) : 0.00459398613277 + 1.0.003666306495
f_hat( 59) : 0.00459398613277 + 1.0.003666306495
f_hat( 60) : 0.00460195686407 + 1.0.003377299083
f_hat( 61) : 0.004602936686 + 1.0.003267349920
f_hat( 62) : 0.0046038352454 + 1.0.003159384834
f_hat( 63) : 0.004603831809 + 1.0.00310718722084
f_hat( 64) : 0.004603831809 + 1.0.00310718722084
f_hat( 65) : 0.004606166207 + 1.0.002848724237
f_hat( 66) : 0.004607951453 + 1.0.002749195844
f_hat( 67) : 0.0046088470439 + 1.0.002749195844
f_hat( 68) : 0.0046088470439 + 1.0.002749195844
f_hat( 69) : 0.0046099313991 + 1.0.002401242376
f_hat( 70) : 0.0046099623338 + 1.0.002368493896
f_hat( 71) : 0.0046101831809 + 1.0.002368493896
f_hat( 72) : 0.0046101831809 + 1.0.002368493896
f_hat( 73) : 0.004611213902 + 1.0.002098789602
f_hat( 74) : 0.004611687651 + 1.0.002011483188
f_hat( 75) : 0.004612135288 + 1.0.001925358824
f_hat( 76) : 0.004612135288 + 1.0.001925358824
f_hat( 77) : 0.004612505586 + 1.0.0017563933802
f_hat( 78) : 0.004613322270 + 1.0.001673430169
f_hat( 79) : 0.004613685956 + 1.0.001593401242
f_hat( 80) : 0.0046141279998 + 1.0.001429527154
f_hat( 81) : 0.00461414729998 + 1.0.001356376761
f_hat( 82) : 0.0046146229013 + 1.0.001356376761
f_hat( 83) : 0.004614694998 + 1.0.001271556394
f_hat( 84) : 0.00461469498 + 1.0.001193399701
f_hat( 85) : 0.00461501232270 + 1.0.0010953597
f_hat( 86) : 0.004615684522 + 1.0.0010389399545
f_hat( 87) : 0.004615806424 + 1.0.000962546465
f_hat( 88) : 0.004615931752 + 1.0.000886637624
f_hat( 89) : 0.0046160132270 + 1.0.0008235333
f_hat( 90) : 0.0046161142898 + 1.0.008736153655
f_hat( 91) : 0.004616451988 + 1.0.000661491306
f_hat( 92) : 0.004616474537 + 1.0.000587162568
f_hat( 93) : 0.0046166046886 + 1.0.000527155705
f_hat( 94) : 0.004616744886 + 1.0.000439351577
f_hat( 95) : 0.004616853089 + 1.0.000365793564
f_hat( 96) : 0.004616916862 + 1.0.000292417482
f_hat( 97) : 0.0046170180433 + 1.0.0002304551
f_hat( 98) : 0.004617081043 + 1.0.0001446064130
f_hat( 99) : 0.004617622732 + 1.0.000073914015
f_hat(100) : 0.004617629771 + 10.0000000000000000
Plot saved to plots/second-func-plot.png
```

.....

End of solution to question 5 :)

teddy@cabin: rust> ./hw3

teddy@cabin rust> ./hw3

numerical@nvin:~\$ vim 2:nvim 2:nvim 3:zsh*

"cabin" 22:45 27-Mar-25

Figure 7: Screenshot of result of code for calculating fourier coefficients using FFT.

```
43 use plotters::prelude::*;
44 use rustfft::num_complex::Complex_FftPlanner;
45 use std::f64::consts::(PI, SQR_T_2);
46
47 fn plot_functions<Fn>(fns: &[Fn], x_range: Range<f64>, y_range: Range<f64>) {
48     let mut chart = ChartBuilder::on(&x_range)
49         .caption("Function Comparison", ("sans-serif", 30).into_font())
50         .margin(10)
51         .x_labels("x")
52         .y_labels("y")
53         .x_desc("f(x) or g(x)*")
54         .y_desc("f(x) or g(x)*")
55         .drawn();
56
57     let points = Vec::from_fn(x_range.step(), |x| fns[0](x));
58
59     for i in 1..fns.len() {
60         let points = points.map(|x| fns[i](x));
61         drawn!(&points);
62     }
63
64     drawn!(&points);
65 }
```

Figure 8: Screenshot of code for plotting function and reconstruction of function.

```
Thursday March 27, 22:32  
to numerical  
  
19 |     .y_labels(10)  
20 |     .x_desc("x")  
21 |     .y_desc("f(x) or g(x)")  
22 |     .drawn()?  
23 |  
24 | let points: Vec<f64, f64> = (0..=sample_points)  
25 |     .map(|i| {  
26 |         let x =  
27 |             x_range.start + (x_range.end - x_range.start) * (i as f64 / sample_points as f64);  
28 |         (x, func(x))  
29 |     })  
30 |     .collect();  
31 |  
32 | let points2: Vec<f64, f64> = (0..=sample_points)  
33 |     .map(|i| {  
34 |         let x =  
35 |             x_range.start + (x_range.end - x_range.start) * (i as f64 / sample_points as f64);  
36 |         (x, reconstruct_function(&fourier_coeffs, 200, x, 5))  
37 |     })  
38 |     .collect();  
39 |  
40 | chart  
41 |     draw series(LineSeries::new(points1.iter().map(|&(x, y)| (x, y)), &RED))?  
42 |     .label("(x)")  
43 |     .legend([(x, y)] PathElement::new(vec![(x, y), (x + 20, y)], &RED));  
44 |  
45 | chart  
46 |     draw series(LineSeries::new(points2.iter().map(|&(x, y)| (x, y)), &BLUE))?  
47 |     .label("g(x)")  
48 |     .legend([(x, y)] PathElement::new(vec![(x, y), (x + 20, y)], &BLUE));  
49 |  
50 | chart  
51 |     .configure_series_labels()  
52 |     .background_style(&WHITE.mix(0.8))  
53 |     .border_style(&BLACK)  
54 |     .drawn()?  
55 |  
56 | root.present()?  
57 |  
58 | println!("Plot saved to {}", filename);  
59 | Ok(())  
60 |}  
61 |  
62 | fn func1(x: f64) -> f64 {  
63 |     (5.0 * x).sin().powi(2)  
64 | }  
65 |  
66 | fn func2(x: f64) -> f64 {  
67 |     (3.0 * x).cos().powi(3) + (SORT_2 * x).cos()  
68 | }  
69 |  
70 | fn reconstruct_function(fourier_coeffs: &[Complex<f64>], n: usize, x: f64, m: i32) -> f64 {  
71 |     let mut sum = 0.0;  
72 |     for j in ..m.m {  
73 |         let mut sum = 0.0;
```

Figure 9: Screenshot of code for plotting function and reconstruction of function.

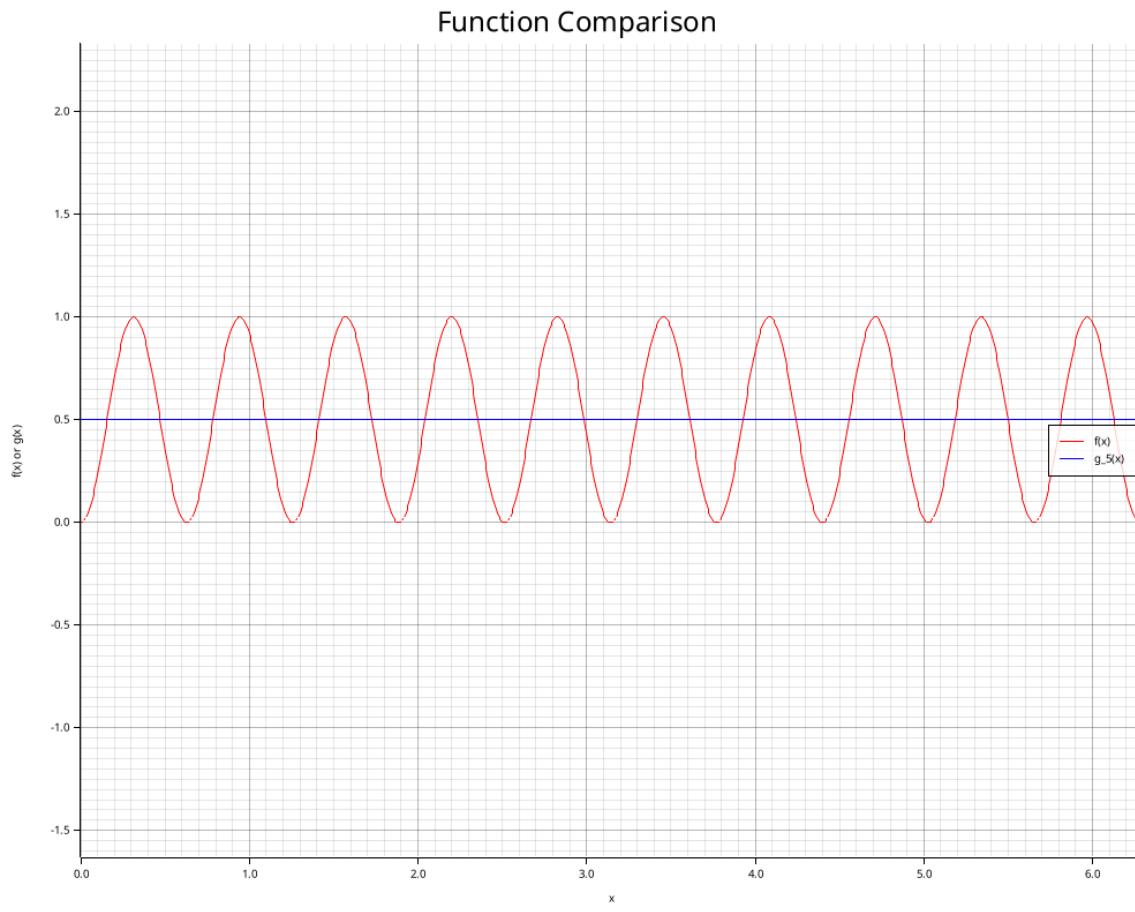


Figure 10: Screenshot of plot of first function and reconstruction for comparison (f vs g_5).

Function Comparison

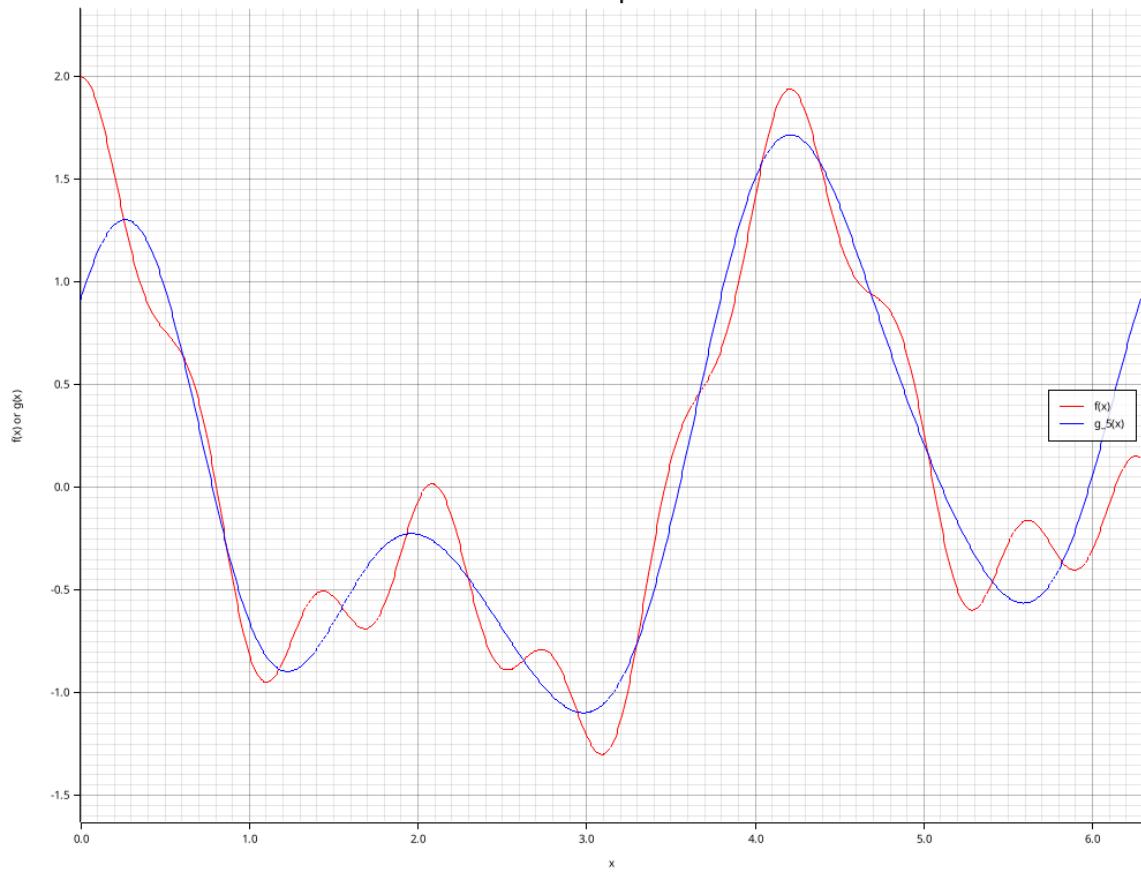


Figure 11: Screenshot of plot of first function and reconstruction for comparison (f vs g_5).

Problem 6: Please see attached PDF with all written solutions for questions 1-4 and 6-7.

Problem 7: Please see attached PDF with all written solutions for questions 1-4 and 6-7.

Problem 8:

```
Thursday March 27, 23:12
ls
~/Documents/Academics/Math/Numerical/
math
main.rs
q5.rs
q8.rs

use std::f64::consts::PI;
fn comp_trap_rule(f: impl Fn(f64) -> f64, a: f64, b: f64, n: usize) -> f64 {
    let h = (b - a) / n as f64;
    let mut sum = 0.5 * (f(a) + f(b));
    for i in 1..n {
        let x = a + (i as f64) * h;
        sum += f(x);
    }
    h * sum
}
pub fn main() {
    let f1 = |x: f64| (-x.powi(2) / 2.0).exp();
    let f2 = |x: f64| 1.0 / (2.0 + x.sin());
    let a1 = 0.0;
    let b1 = 1.0;
    let a2 = 0.0;
    let b2 = 2.0 * PI;
    println!("a) integral from 0 to 1 of e^(-x^2 / 2) dx");
    for n in [128, 256, 512] {
        let result = comp_trap_rule(f1, a1, b1, n);
        println!("n = {} : {:.16}, n, result");
    }
    println!("b) integral from 0 to 2pi of 1/(2 + sin(x)) dx");
    for n in [128, 256, 512] {
        let result = comp_trap_rule(f2, a2, b2, n);
        println!("n = {} : {:.16}, n, result");
    }
}

neo-tree filesystem [1] Normal ↵ hw3 ↵ ⌂ ~ ~/Documents/Academics/Math/Numerical/numerical_analysis_math4640/solutions/rust/hw3/src/q8.rs ↵ rust utf-8[unix] 862B 14:1
Running target/debug/hw3 q8
...MATH 4640, Numerical Analysis, Homework 3...
.....
Running solution to question 8...
a) integral from 0 to 1 of e^(-x^2 / 2) dx
n = 128: 0.8556213069
n = 256: 0.8556236206
n = 512: 0.8556236286
b) integral from 0 to 2pi of 1/(2 + sin(x)) dx
n = 128: 3.6275987285
n = 256: 3.6275987285
[numerical@o:nvim* 1:zsh- 2:zsh 3:zsh]
```

Figure 12: Screenshot of the code for implementing the composite trapezoidal rule.

```
Thursday March 27, 23:11
teddy@cabin rust/hw3 (hw3) » cargo run q8
Compiling hw3 v0.1.0 (/home/teddy/Documents/Academics/Math/Numerical/numerical_analysis_math4640/solutions/rust/hw3)
warning: function `round_coefficients` is never used
--> src/math/approx.rs:1:6
1 pub fn round_coefficients(d: &[f64], tol: f64) -> Vec<f64> {
   |
   = note: #![warn(dead_code)] on by default
warning: function `ndd` is never used
--> src/math/interpolation.rs:21:8
2 pub fn ndd(x: &[f64], y: &[f64]) -> Vec<f64> {
   |
   = note: #![warn(dead_code)] on by default
warning: function `interpolate_polynomial` is never used
--> src/math/interpolation.rs:43:8
3 pub fn interpolate_polynomial(x: &[f64], d: &[f64], t: &f64, n: &usize) -> f64 {
   |
   = note: this `Result` may be an `Err` variant, which should be handled
   = note: #![warn(unused must_use)] on by default
help: use `let _ = ...` to ignore the resulting value
154 let _ = plot_functions(func, &b64buffer, filename);
   |
   = note: this `Result` may be an `Err` variant, which should be handled
   = note: #![warn(unused must_use)] on by default
   help: use `let _ = ...` to ignore the resulting value
155 let _ = plot_functions(func, &b64buffer, filename);
   |
   = note: this `Result` may be an `Err` variant, which should be handled
   = note: #![warn(unused must_use)] on by default
help: use `let _ = ...` to ignore the resulting value
warning: 'hw3' (bin "hw3") generated 4 warnings
  Failed to add 'dev' profile [unoptimized + debuginfo] target(s) in 1.845s
  Running target/debug/hw3 q8
...MATH 4640, Numerical Analysis, Homework 3...
.....
Running solution to question 8...
a) integral from 0 to 1 of e^(-x^2 / 2) dx
n = 128: 0.8556213069114667
n = 256: 0.8556236206481403
n = 512: 0.8556236286477077
b) integral from 0 to 2pi of 1/(2 + sin(x)) dx
n = 128: 3.6275987284684348
n = 256: 3.6275987284684339
n = 512: 3.6275987284684386
.....
End of solution to question 8 :)
```

Figure 13: Screenshot of the results on each given case for implementing the composite trapezoidal rule.

1 Written Solutions

HW 3

Question 1 Let $\{P_i(x)\}_{i=0}^{\infty}$ be a sequence of orthogonal polynomials and let x_0, x_1, \dots, x_k be the $k+1$ distinct zeros of $P_{k+1}(x)$.

Prove that the Lagrange polynomials $l_i(x) = \prod_{t=0, t \neq i}^{k+1} (x - x_t) / (x_i - x_t)$ for these points are orthogonal $\int l_i l_j dx = 0$ to each other.

Hint: Show that for $i \neq j$, $\int l_i l_j dx = P_{k+1}(x) g(x)$, where $g(x)$ is some polynomial of degree $\leq k$.

Solution

$$l_i(x) = \prod_{\substack{t=0 \\ t \neq i}}^{k+1} \frac{x - x_t}{x_i - x_t} \quad \text{and} \quad l_j(x) = \prod_{\substack{s=0 \\ s \neq j}}^{k+1} \frac{x - x_s}{x_j - x_s}$$

$$l_i(x_t) = \begin{cases} 1 & t = i \\ 0 & t \neq i \end{cases} \quad l_j(x_s) = \begin{cases} 1 & s = j \\ 0 & s \neq j \end{cases}$$

$$\deg(l_i(x)) = \deg(l_j(x)) = k, \quad \deg(l_i(x) l_j(x)) \leq 2k$$

$$l_i(x) l_j(x) = \prod_{\substack{t=0 \\ t \neq i}}^{k+1} \frac{x - x_t}{x_i - x_t} \prod_{\substack{s=0 \\ s \neq j}}^{k+1} \frac{x - x_s}{x_j - x_s}$$

$$l_i(x_t) l_j(x_s) = \begin{cases} (1)(0) & t = i \\ (0)(1) & t = j \\ (0)(0) & t \neq i \text{ and } t \neq j \end{cases}$$

$$\text{Thus, } l_i(x_t) l_j(x_s) = 0 \quad x_0, \dots, x_k$$

So $l_i(x) l_j(x)$ is a degree $\leq 2k$ polynomial with $k+1$ distinct roots at x_0, \dots, x_k .

This means, for some $k+1$ degree or less polynomial $g(x)$

$$l_i(x) l_j(x) = P_{k+1}(x) g(x)$$

But $P_{k+1}(x)$ is orthogonal and $k+1$ degree, so $\langle P_{k+1}, g \rangle = 0$

$$\text{Hence } \langle l_i, l_j \rangle = 0$$

Question 2

Approximation by Orthogonal Projection

The best polynomial $p(x)$ is given by projecting onto the basis of:

$$\text{- Legendre polynomials for } \langle f, g \rangle = \int_{-1}^1 f(x)g(x)dx$$

$$\text{- Chebyshev polynomials for } \langle f, g \rangle = \int_{-1}^1 \frac{f(x)g(x)}{L(-x^2)^{1/2}} dx$$

$$\text{- Hermite polynomials for } \langle f, g \rangle = \int_{-\infty}^{\infty} f(x)g(x)e^{-x^2}dx$$

For up to degree 2 with our objective being to minimize

$$\int_{-1}^1 (\cos(\pi x) - p(x))^2 dx$$

we note this corresponds to $\langle f, g \rangle = \int_{-1}^1 f(x)g(x)dx$ inner product.

$$p_2(x) = \sum_{i=0}^2 \frac{\langle \cos(\pi x), L_i \rangle}{\langle L_i, L_i \rangle} L_i(x)$$

$$L_0 = 1, \quad L_1(x) = x, \quad L_2(x) = \frac{3x^2 - 1}{2} = \frac{3}{2}x^2 - \frac{1}{2}$$

$$\text{As } L_{n+1}(x) = \frac{(2n+1)xL_n(x) - nL_{n-1}(x)}{n+1} \text{ is Legendre polynomial general form.}$$

$$\text{Further, } \langle L_i, L_i \rangle = \frac{2}{2i+1}. \text{ Thus,}$$

$$p_2(x) = \frac{\langle \cos(\pi x), 1 \rangle}{2} + \frac{\langle \cos(\pi x), x \rangle}{3} x$$

$$+ \frac{\langle \cos(\pi x), \frac{3}{2}x^2 - \frac{1}{2} \rangle}{5} \left(\frac{3}{2}x^2 - \frac{1}{2} \right)$$

$$\cdot \langle \cos(\pi x), 1 \rangle = \int_{-1}^1 \cos(\pi x) dx = \frac{1}{\pi} \sin(\pi x) \Big|_{-1}^1$$

$$= \frac{1}{\pi} (\sin(\pi) - \sin(-\pi)) = \frac{1}{\pi} (\sin(\pi) + \sin(\pi)) = 0$$

$$\cdot \langle \cos(\pi x), x \rangle = \int_{-1}^1 \cos(\pi x) x dx = 0 \quad \text{(steps below)}$$

$$\int u dv = uv - \int v du \quad u = x \quad v = \frac{1}{\pi} \sin(\pi x)$$

$$du = dx \quad dv = \cos(\pi x) dx$$

$$\int x \cos(\pi x) dx = \frac{x}{\pi} \sin(\pi x) - \frac{1}{\pi} \int \sin(\pi x) dx$$

$$\int_{-1}^1 x \cos(\pi x) dx = \frac{x}{\pi} \sin(\pi x) \Big|_{-1}^1 + \frac{1}{\pi} \int_{-1}^1 \cos(\pi x) dx$$

$$= \frac{1}{\pi^2} (\cos(\pi) - \cos(-\pi)) = \frac{1}{\pi^2} (0) = 0$$

$$\cdot \langle \cos(\pi x), \frac{3}{2}x^2 - \frac{1}{2} \rangle = \frac{3}{2} \int_{-1}^1 x^2 \cos(\pi x) dx - \frac{1}{2} \int_{-1}^1 \cos(\pi x) dx$$

$$\cdot -\frac{1}{2} \int_{-1}^1 \cos(\pi x) dx = 0$$

$$\cdot \frac{3}{2} \int_{-1}^1 x^2 \cos(\pi x) dx = \frac{3}{2} \left(-\frac{4}{\pi^2} \right) = -\frac{12}{2\pi^2} = -\underline{\underline{6\pi^{-2}}}$$

$$u = x^2 \quad v = \frac{1}{\pi} \sin(\pi x)$$

$$du = 2x dx \quad dv = \cos(\pi x) dx$$

$$\int_{-1}^1 x^2 \cos(\pi x) dx = \frac{x^2}{\pi} \sin(\pi x) \Big|_{-1}^1 - \frac{1}{\pi} \int_{-1}^1 \sin(\pi x) 2x dx = -\frac{1}{\pi} \int_{-1}^1 \sin(\pi x) 2x dx$$

$$\Rightarrow -\frac{1}{\pi} \int_{-1}^1 \sin(\pi x) 2x dx = -\frac{2}{\pi} \left(\frac{\sin(\pi x) - \pi x \cos(\pi x)}{\pi^2} \Big|_{-1}^1 \right) = -\frac{2}{\pi} \left(\frac{2}{\pi} \right) = -\underline{\underline{4\pi^{-2}}}$$

$$\cdot \langle \cos(\pi x), 1 \rangle = 0$$

$$\cdot \langle \cos(\pi x), x \rangle = 0$$

$$\cdot \langle \cos(\pi x), \frac{3}{2}x^2 - \frac{1}{2} \rangle = -6\pi^{-2}$$

$$p_2(x) = \frac{-6\pi^{-2}}{\frac{2}{5}} \left(\frac{3}{2}x^2 - \frac{1}{2} \right) = \frac{-30}{2\pi^2} \left(\frac{3}{2}x^2 - \frac{1}{2} \right)$$

$$= -15\pi^{-2} \left(\frac{3}{2}x^2 - \frac{1}{2} \right) = \frac{-45\pi^{-2}}{2} x^2 + \frac{15\pi^{-2}}{2}$$

$$p_2(x) = -\frac{45}{2\pi^2} x^2 + \frac{15}{2\pi^2} . \quad \square$$

Question 3 If $f(x)$ is a 2π periodic function, prove that $g_\alpha(x) = f(x+\alpha)$ is also 2π -periodic.

Proof.

We want to show $g_\alpha(x+2\pi) = g_\alpha(x) \quad \forall x$.

$$g_\alpha(x+2\pi) = f((x+2\pi)+\alpha) = f(x+\alpha+2\pi)$$

Since f is 2π periodic, $f((x+\alpha)+2\pi) = f(x+\alpha)$

Hence, $g_\alpha(x+2\pi) = g_\alpha(x)$, as $f(x+\alpha) = g_\alpha(x)$.

Therefore, $g_\alpha(x)$ is also 2π -periodic. \square

$$f(j) = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(x) e^{-ijx} dx \quad \hat{g}_\alpha(j) = \frac{1}{2\pi} \int_{-\pi}^{\pi} g_\alpha(x) e^{-ijx} dx$$

$$\hat{g}_\alpha(j) = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(x+\alpha) e^{-ijx} dx$$

$$u = x + \alpha \quad du = dx \quad \text{so} \quad 0 \mapsto \alpha, \quad 2\pi \mapsto 2\pi + \alpha$$

$$\hat{g}_\alpha(j) = \frac{1}{2\pi} \int_{\alpha}^{2\pi+\alpha} f(u) e^{-ij(u-\alpha)} du \quad x = u - \alpha$$

$$f(u) \text{ is } 2\pi\text{-periodic, so } \frac{1}{2\pi} \int f(u) e^{-ij(u-\alpha)} du = \frac{1}{2\pi} \int f(u) e^{-ij(u-\alpha)} du$$

$$\text{And thus, } \hat{g}_\alpha(j) = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(u) e^{-ij(u-\alpha)} e^{ij\alpha} du$$

which is:

$$\hat{g}_\alpha(j) = \frac{e^{ij\alpha}}{2\pi} \int_{-\pi}^{\pi} f(u) e^{iju} du$$

$$\text{Thus, } \hat{g}_\alpha(j) = e^{ij\alpha} f(j). \quad \square$$

Question 4 Verify that the 2π -periodic function $f(x)$ whose values on $[0, 2\pi]$ are given by

$$f(x) = \begin{cases} (\frac{x}{\pi})^2 - x/\pi & 0 \leq x < \pi \\ (x-\pi)\pi - \left(\frac{x-\pi}{\pi}\right)^2 & \pi \leq x < 2\pi \end{cases}$$

is continuous and has a continuous first derivative (as a 2π -periodic) but has jumps in the second derivative. Then construct the spectrum of $f(x)$ and show that it decays like j^{-3} , $j \rightarrow \infty$.

Solution

$$f'(x) = \begin{cases} \frac{2x}{\pi^2} - \frac{1}{\pi}, & x \in [0, \pi) \\ \frac{1}{\pi} - \frac{2(x-\pi)}{\pi^2}, & x \in [\pi, 2\pi) \end{cases}$$

$$f''(x) = \begin{cases} \frac{2}{\pi^2}, & x \in [0, \pi) \\ -\frac{2}{\pi^2}, & x \in [\pi, 2\pi) \end{cases}$$

- $\lim_{x \rightarrow \pi^-} f(x) = \left(\frac{\pi}{\pi}\right)^2 - \frac{\pi}{\pi} = 1 - 1 = 0$

- $\lim_{x \rightarrow \pi^+} f(x) = \frac{\pi-\pi}{\pi} - \left(\frac{\pi-\pi}{\pi}\right)^2 = 0$

- $\lim_{x \rightarrow \pi^-} f(x) = \lim_{x \rightarrow \pi^+} f(x)$, f is continuous over $[0, 2\pi]$

- $\lim_{x \rightarrow \pi^-} f'(x) = \frac{2\pi}{\pi^2} - \frac{1}{\pi} = \frac{1}{\pi}$

- $\lim_{x \rightarrow \pi^+} f'(x) = \frac{1}{\pi} - 0 = \frac{1}{\pi}$

- $\lim_{x \rightarrow \pi^-} f'(x) = \lim_{x \rightarrow \pi^+} f'(x)$, F' is continuous over $[0, 2\pi]$

- $\lim_{x \rightarrow \pi^-} f''(x) = \frac{2}{\pi^2} \neq \lim_{x \rightarrow \pi^+} f''(x) = -\frac{2}{\pi^2}$ f'' is not contn. over $[0, 2\pi]$.

$$\hat{f}(j) = \frac{1}{2\pi} \int_0^{2\pi} f(x) e^{-ijx} dx$$

$$= \frac{1}{2\pi} \left[\int_0^{\pi} \left(\frac{x}{\pi} \right)^2 e^{-ijx} - \frac{x}{\pi} e^{-ijx} dx + \int_{\pi}^{2\pi} \frac{x-\pi}{\pi} e^{-ijx} - \frac{(x-\pi)^2}{\pi^2} e^{-ijx} dx \right]$$

Or if using $f(x) = \sum_{j=1}^n a_j \sin jx + \sum_{j=0}^n b_j \cos jx$
form instead, where:

$$a_j = \langle f, \sin jx \rangle$$

$$b_j = \langle f, \cos jx \rangle$$

$$a_j = \frac{1}{2\pi} \int_0^{2\pi} f(x) \sin jx dx = \frac{1}{2\pi} \left[\int_0^{\pi} \left(\frac{x^2}{\pi^2} - \frac{x}{\pi} \right) \sin jx dx + \int_{\pi}^{2\pi} \left(\frac{x-\pi}{\pi} - \frac{(x-\pi)^2}{\pi^2} \right) \sin jx dx \right]$$

$$b_j = \frac{1}{2\pi} \int_0^{2\pi} f(x) \cos jx dx = \frac{1}{2\pi} \left[\int_0^{\pi} \left(\frac{x^2}{\pi^2} - \frac{x}{\pi} \right) \cos jx dx + \int_{\pi}^{2\pi} \left(\frac{x-\pi}{\pi} - \frac{(x-\pi)^2}{\pi^2} \right) \cos jx dx \right]$$

$f \in C^1[0, 2\pi]$ since $f^{(2)}(x)$ is not continuous over $[0, 2\pi]$ but $f, f^{(1)}$ are.

$$\text{Thus, } |\hat{f}(j)| = O(|j|^{-2}) \quad \square$$

Question 6

First need to determine the orthogonal polynomials associated with $w(x) = -x$ over $[-1, 0]$.

$$\text{Satisfy: } \int_{-1}^0 P_m(x) P_n(x) w(x) dx = 0 \quad \text{for } m \neq n.$$

These are the Legendre polynomials shifted by $x \mapsto -2x - 1$. Let $\tilde{P}_n(x)$ be non-shifted Legendre poly. $[-1, 0] \rightarrow [-1, 1]$

$$\text{So, } P_0(x) = 1, \quad P_1(x) = \tilde{P}_1(2x+1) = 2x+1$$

$$P_2(x) = \tilde{P}_2(2x+1) = \frac{1}{2}(3(2x+1)^2 - 1) = \frac{3}{2}(4x^2 + 4x + 1) - \frac{1}{2}$$

$$= 6x^2 + 6x + 1$$

(n=1) One-Point Quadrature

$P_1(x) = 2x+1$. The quadrature node is the root of $P_1(x)$, which is $x_0 = -\frac{1}{2}$.

$$\text{The weight is } A_0 = \int_{-1}^0 (-x) \lambda_0(x) dx = \int_{-1}^0 -x dx = \left[-\frac{x^2}{2} \right]_{-1}^0 = \frac{1}{2}$$

The one-point Gaussian quadrature rule is

$$I \approx \frac{1}{2} f(x_0) = \frac{1}{2} f(-\frac{1}{2})$$

(n=2) Two-Point Quadrature

$$P_2(x) = 6x^2 + 6x + 1. \text{ Solving } 6x^2 + 6x + 1 = 0 \text{ we find,}$$

$$x = \frac{-6 \pm \sqrt{36 - 24}}{12} = \frac{-6 \pm \sqrt{12}}{12} = \frac{-6 \pm 2\sqrt{3}}{12} = -\frac{1}{2} \pm \frac{1}{6}\sqrt{3}$$

$$x_0 = -\frac{1}{2} - \frac{1}{6}\sqrt{3}, \quad x_1 = -\frac{1}{2} + \frac{1}{6}\sqrt{3}$$

$$A_0 = \int_{-1}^0 -x \lambda_0(x) dx = \int_{-1}^0 -x \frac{x-x_1}{x_0-x_1} dx = \frac{1}{x_0-x_1} \int_{-1}^0 -x^2 + xx_1 dx = \frac{1}{x_0-x_1} \left[\frac{-x^3}{3} + \frac{x^2}{2} x_1 \right]_{-1}^0$$

$$A_0 = \frac{\frac{1}{3} + \frac{1}{2}x_1}{x_0-x_1}$$

$$A_1 = \int_{-1}^0 -x \lambda_1(x) dx = \int_{-1}^0 -x \frac{x-x_0}{x_1-x_0} dx = \frac{1}{x_1-x_0} \left[\frac{-x^3}{3} + \frac{x^2}{2} x_0 \right]_{-1}^0 = \frac{\frac{1}{3} + \frac{1}{2}x_0}{x_1-x_0}$$

The Two-Point Quadrature Rule is thus,

$$A_0 f(x_0) + A_1 f(x_1) = \frac{\frac{1}{3} + \frac{1}{2}x_1}{x_0 - x_1} f(x_0) + \frac{\frac{1}{3} + \frac{1}{2}x_0}{x_1 - x_0} f(x_1)$$

$$\therefore \frac{1}{3} + \frac{1}{2}x_1 = \frac{1}{3} - \frac{1}{4} + \frac{\sqrt{3}}{12} = \frac{1 + \sqrt{3}}{12}$$

$$\therefore \frac{1}{3} + \frac{1}{2}x_0 = \frac{1}{3} - \frac{1}{4} - \frac{\sqrt{3}}{12} = \frac{1 - \sqrt{3}}{12}$$

$$\therefore x_0 - x_1 = \left(-\frac{1}{2} - \frac{\sqrt{3}}{6}\right) - \left(-\frac{1}{2} + \frac{\sqrt{3}}{6}\right) = -\frac{\sqrt{3}}{3}$$

$$\therefore x_1 - x_0 = \left(-\frac{1}{2} + \frac{\sqrt{3}}{6}\right) - \left(-\frac{1}{2} - \frac{\sqrt{3}}{6}\right) = \frac{\sqrt{3}}{3}$$

Final Answer

$$\frac{1 + \frac{\sqrt{3}}{12}}{-\frac{\sqrt{3}}{3}} f\left(-\frac{1}{2} - \frac{\sqrt{3}}{6}\right) + \frac{1 - \frac{\sqrt{3}}{12}}{\frac{\sqrt{3}}{3}} f\left(-\frac{1}{2} + \frac{\sqrt{3}}{6}\right)$$

2

Question 7

The corrected trapezoidal rule is

$$I(f) = \int_a^b f(x) dx \approx \frac{1}{2} (b-a) (f(a)+f(b)) + \frac{(b-a)^2}{12} (f'(a)-f'(b))$$

For evenly spaced points we let $x_i = x_0 + ih$
for x_0, \dots, x_N and $h = \frac{b-a}{N}$.

So $x_{i+1} - x_i = h, \forall i = 0, \dots, N-1$. Now, we want to

$$I(f) = \sum_{i=0}^{N-1} \int_{x_i}^{x_{i+1}} f(x) dx \quad \text{compute}$$

But for the corrected trapezoidal rule,

$$\begin{aligned} \int_{x_i}^{x_{i+1}} f(x) dx &\approx \frac{1}{2} (x_{i+1} - x_i) (f(x_i) + f(x_{i+1})) + \frac{(x_{i+1} - x_i)^2}{12} (f'(x_i) - f'(x_{i+1})) \\ &= \frac{1}{2} h (f(x_i) + f(x_{i+1})) + \frac{h^2}{12} (f'(x_i) - f'(x_{i+1})) \end{aligned}$$

So,

$$\begin{aligned} I(f) &\approx \sum_{i=0}^{N-1} \left[\frac{h}{2} (f(x_i) + f(x_{i+1})) + \frac{h^2}{12} (f'(x_i) - f'(x_{i+1})) \right] \\ &= \frac{h}{2} \sum_{i=0}^{N-1} [f(x_i) + f(x_{i+1})] + \frac{h^2}{12} \sum_{i=0}^{N-1} [f'(x_i) - f'(x_{i+1})] \end{aligned}$$

$$\text{Answer} \Rightarrow = \frac{h}{2} (f(a) + f(b)) + h \sum_{i=1}^{N-1} f(x_i) + \frac{h^2}{12} (f'(a) - f'(b)) \quad \star$$

$$\text{Since } \sum_{i=0}^{N-1} [f'(x_i) - f'(x_{i+1})]$$

$$\begin{aligned} &= f'(x_0) - f'(x_1) + f'(x_1) - f'(x_2) + \dots - f'(x_{N-1}) + f'(x_{N-1}) - f'(x_N) \\ &= f'(x_0) - f'(x_N) = f'(a) - f'(b) \quad \blacksquare \end{aligned}$$