

Function Approximation

Mark A. Austin

University of Maryland

austin@umd.edu

ENCE 201, Fall Semester 2023

September 25, 2023

Overview

- 1 **Function Approximation**
 - Polynomial Approximation, Fourier Series, Machine Learning
- 2 **Taylor Series**
 - Motivating Idea
 - Taylor Series and Maclaurin Expansions
 - Remainder Function
 - Ratio Test and Interval of Convergence
- 3 **Solved Problems**
 - Example 1: Taylor Series for e^x about $x = 0$.
 - Example 2: Taylor Series for $\cos(x)$ about $x = 0$.
- 4 **Fourier Series (and Fourier Integral)**
- 5 **Python Code Listings**

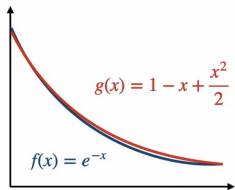
Function Approximation

Motivating Ideas

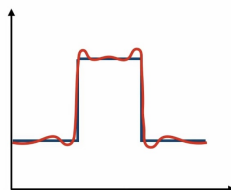
Function Approximation

A **function approximation** asks us to **select a function**, $g(x)$, among a well-defined set of options that approximates – **closely matches** – a second function, $f(x)$, in a **task-specific way**.

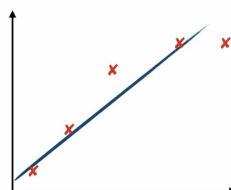
Approximation Examples: Many approaches ...



Polynomial Approximation



Fourier Series



Machine Learning

Strategy 1: Polynomial Approximation

Polynomial Approximation

Replace function $f(x)$ by a **simpler polynomial approximation** $g(x)$.
Then, use $g(x)$ in computations instead of $f(x)$.

Example 1: Replace $y = f(x) = e^{-x}$ by a quadratic approximation:

$$f(x) = e^{-x} \quad \longrightarrow \quad g(x) = 1 - x + \frac{x^2}{2}. \quad (1)$$

Example 2: Replace $y = \sin(x)$ on $x \in [0, \pi]$ by a quadratic approximation:

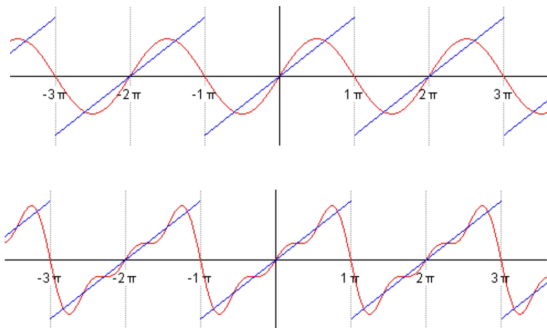
$$f(x) = \sin(x) \quad \longrightarrow \quad g(x) = \frac{4x}{\pi^2} [\pi - x]. \quad (2)$$

Strategy 2: Fourier Series

Fourier Series

A Fourier series is an expansion of a **periodic function** $f(x)$ in terms of an **infinite sum** of **trigonometric** (i.e., sines and cosines) and/or **exponential functions**.

Example 1: Progressive refinement of sawtooth function:

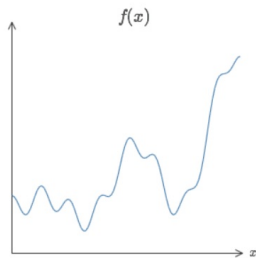


Strategy 3: Machine Learning

Neural Network

Use observable (or experimentally measured) data to train a **neural network** to capture (or estimate) **input-to-output functionality**.

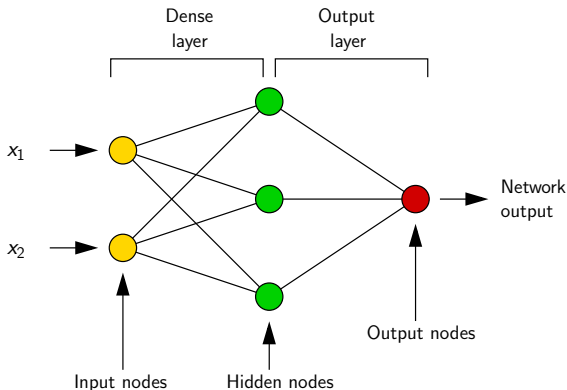
- Neural networks are **universal function approximators**, no matter how complex:
- Neural **network architectures** are **highly scalable** and **flexible**.



Caveat: Very large neural networks may be close to impossible to train and generalize correctly → AI chips.

Strategy 3: Machine Learning

Example 1: Neural Network with One Hidden Layer:



Data inputs x_1 and x_2 are transformed into a prediction $f(x_1, x_2)$.

Strategy 3: Machine Learning

Example 2: Learn how to [classify univariate time series](#) as belonging to one of six categories:

Data

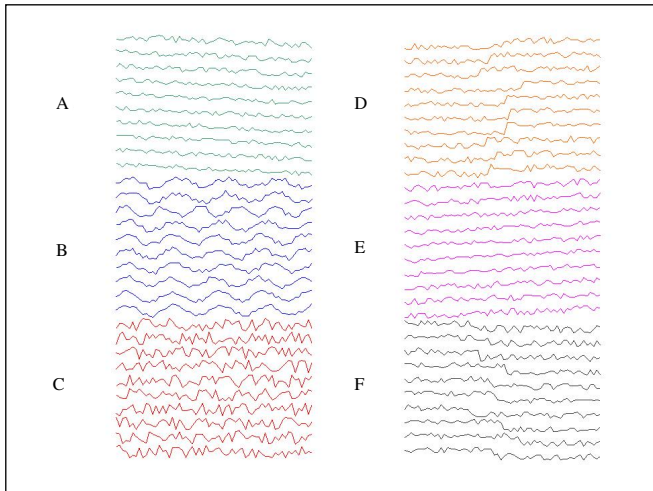
- UCI Synthetic Control Chart Time Series Data Set contains 600 sequences of data.
- Partition data: 450 items for training; 150 items for testing.

Six Categories of Datastream

- **A and E:** Decreasing and Increasing Trend.
- **B:** Cyclic.
- **C:** Normal.
- **D and F:** Upward and Downward Shift.

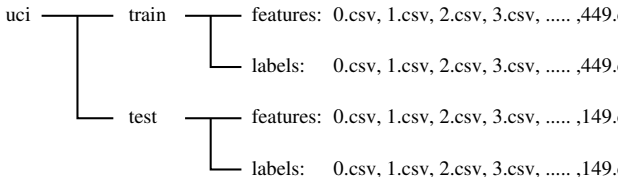
Strategy 3: Machine Learning

Representative Data Streams:



Strategy 3: Machine Learning

Training and Testing Corpus



CSV Data File Format and Label Mappings

Features: 0.csv

20.59 ← line 01
18.51
⋮
16.32
10.72 ← line 60

Labels: 0.csv

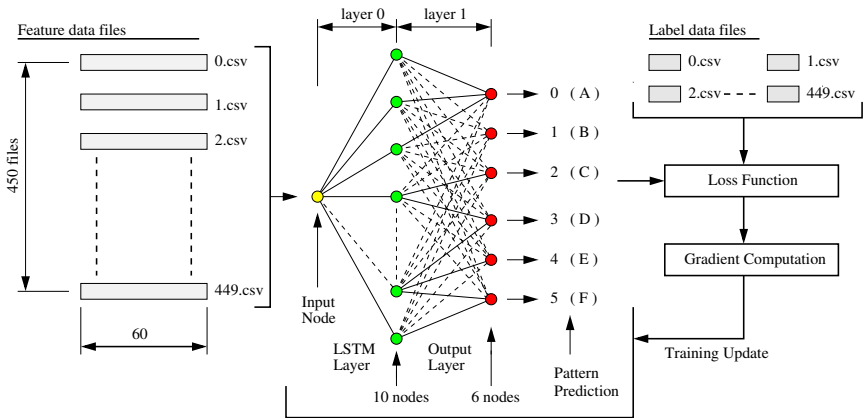
3

Output Mapping

0 ← A
1 ← B
2 ← C
3 ← D
4 ← E
5 ← F

Strategy 3: Machine Learning

RNN Architecture + Sequences of Feature and Label vectors



Strategy 3: Machine Learning

Training the Model for 40 Epochs:

```
int nEpochs = 40;
net.fit(trainData, nEpochs);
```

Evaluation Metrics and Confusion Matrix:

```
Accuracy: 0.8867   Recall: 0.8890   <--- It works!
Precision: 0.8886   F1 Score: 0.8883
```

	0	1	2	3	4	5	

26	0	0	0	0	0	0	0 = 0
0	29	0	0	0	0	0	1 = 1
0	0	15	0	7	0	0	2 = 2
0	0	0	20	0	1	0	3 = 3
0	0	9	0	21	0	0	4 = 4
0	0	0	0	0	22	0	5 = 5

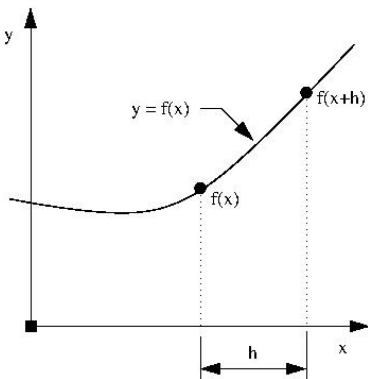
Taylor Series

Taylor Series

(Brook Taylor, 1715)

Motivating Idea

Let $y = f(x)$ be a smooth differentiable function.



Given $f(x)$ and derivatives $f'(a)$, $f''(a)$, $f'''(a)$, etc, the purpose of Taylor's series is to estimate $f(x+h)$ at some distance h from x .

Taylor Series Expansion

Mathematical Expansion.

$$f(x+h) = \sum_{i=0}^{\infty} \frac{f^{(i)}(x)}{(i)!} h^i = f(x) + f'(x)h + \frac{f''(x)}{2!} h^2 + \frac{f'''(x)}{3!} h^3 + \dots \quad (3)$$

For a Taylor series approximation containing $(n+1)$ terms

$$f(x+h) = \sum_{i=0}^{i=n} \frac{f^{(i)}(x)}{(i)!} h^i + R_n(x) \quad (4)$$

The remainder, $R_n(x)$, after truncation is:

$$R_n(x) = \frac{f^{(n+1)}(c(x+h))}{(n+1)!} h^{(n+1)} \quad \text{with} \quad [x \leq c \leq x+h]. \quad (5)$$

Taylor Series Expansion

We can also write:

$$f(x+h) = \sum_{i=0}^{i=n} \frac{f^i(x)}{(i)!} h^n + O(h^{(n+1)}) \quad (6)$$

The big-O notation $O(h^n)$ indicates how quickly the error will change as a function of h .

- $O(0)$ → Magnitude of error is constant, regardless of h .
- $O(h)$ → Magnitude of error proportional to h .
- $O(h^2)$ → Magnitude of error proportional to h squared.

Maclaurin Series Expansion

Maclaurin Series: A **Maclaurin Series** is nothing more than a Taylor series expansion about $a = 0$, i.e.,

$$f(h) = f(0) + \frac{f'(0)}{1!}h + \frac{f''(0)}{2!}h^2 + \frac{f'''(0)}{3!}h^3 + \dots \quad (7)$$

Trigonometric Series

$$\sin(x) = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!} x^{2n+1} = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots$$

$$\cos(x) = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!} x^{2n} = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots$$

Surprisingly, these series converge for all values of x .

Remainder Function

Remainder Function. The formula for $R_n(x)$,

$$R_n(x) = \frac{f^{(n+1)}(c(x+h))}{(n+1)!} h^{(n+1)} \quad \text{with} \quad [x \leq c \leq x+h]. \quad (8)$$

This formula is derived via the “mean value theorem” (see extra slides for details) and simply states there exists a point $c(x+h)$ such that:

$$\frac{f(x+h) - f(x)}{(x+h-x)} = f'(c) \rightarrow f(x+h) = f(x) + f'(c)(h). \quad (9)$$

We can estimate $R_n(x)$ without knowing $c(x+h)$ explicitly.

Ratio Test and Interval of Convergence

For the power series centered about $x = a$,

$$P(a+h) = C_0(a) + C_1(a)h + C_2(a)h^2 + C_3(a)h^3 + C_4(a)h^4 + \dots \quad (10)$$

suppose that:

$$\lim_{n \rightarrow \infty} \left[\frac{|C_n(a)|}{|C_{n+1}(a)|} \right] = R. \quad (11)$$

then:

- If $R = \infty$, then the series converges for all values.
- If $0 < R < \infty$, then the series converges for all $h < R$.
- If $R = 0$, then the series converges only for $h = 0$.

We call R the **radius of convergence**.

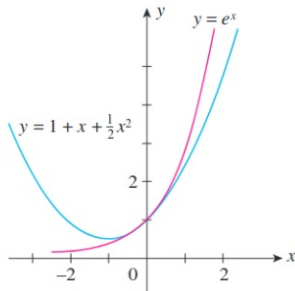
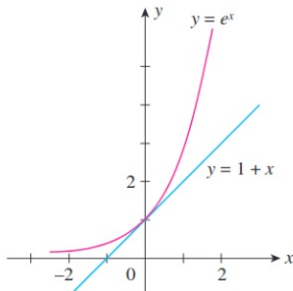
Solved Problems

Example 1: Taylor Series Expansion for e^x

Problem 1. Approximating e^x about $x = 0$.

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \frac{x^5}{5!} + \dots \quad (12)$$

Linear and quadratic approximations:



Example 1: Taylor Series Expansion for e^x

Now lets predict: $e^2 = 2.71828 * 2.71828 = 7.38904$.

No Terms	Numerical Estimate
1	1.0000
2	$1 + 2 \rightarrow 3.0000$
3	$1 + 2 + 4/2! \rightarrow 5.00000$
4	$1 + 2 + 4/2! + 8/3! \rightarrow 6.33333$
5	$1 + 2 + 4/2! + 8/3! + 16/4! \rightarrow 6.99999$

Estimate of Maximum Error: After five terms:

$$R_5(2) \leq \frac{e^{c(2)}}{6!} 2^6 = \frac{e^2}{6!} 2^6 = \frac{7.38904 * 64}{720} = 0.657. \quad (13)$$

The actual error is: 0.389.

Example 1: Taylor Series Expansion for e^x

Test for Convergence. We have:

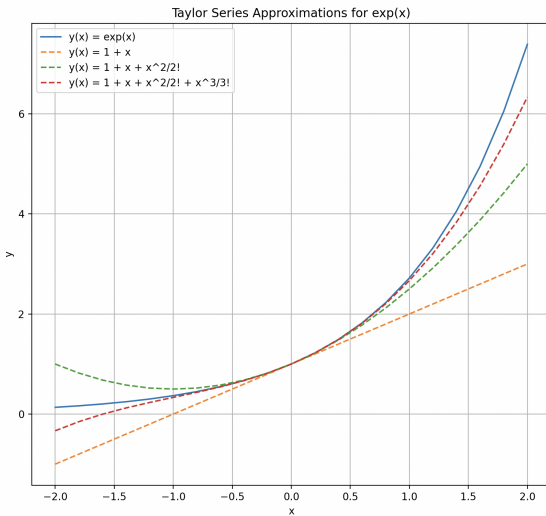
$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} \quad (14)$$

The limit

$$\lim_{n \rightarrow \infty} \left[\frac{|C_n(a)|}{|C_{n+1}(a)|} \right] = \lim_{n \rightarrow \infty} \left[\frac{\frac{1}{n!}}{\frac{1}{(n+1)!}} \right] = \lim_{n \rightarrow \infty} (n+1) = \infty. \quad (15)$$

gives $R = \infty$ and the series converges for all values of x .

Example 1: Taylor Series Expansion for $\exp(x)$



Example 2: Taylor Series Expansion for $\cos(x)$

Example 2. Taylor Series expansion for $\cos(x)$ about $x = 0$.

We have: $f(0) = \cos(0)$, $f^1(0) = -\sin(0)$, and so forth. Hence,

$$\begin{aligned}\cos(x) &= \cos(0) - \frac{\sin(0)}{1!}x + \frac{-\cos(0)}{2!}x^2 + \frac{\sin(0)}{3!}x^3 + \frac{\cos(0)}{4!}x^4 + \dots \\ &= 1 - \frac{x^2}{2!} + \frac{x^4}{4!} + \dots\end{aligned}$$

Can also use the same procedure to show:

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} + \dots \quad (16)$$

Example 2: Taylor Series Expansion for $\cos(x)$

Test for Convergence. We have:

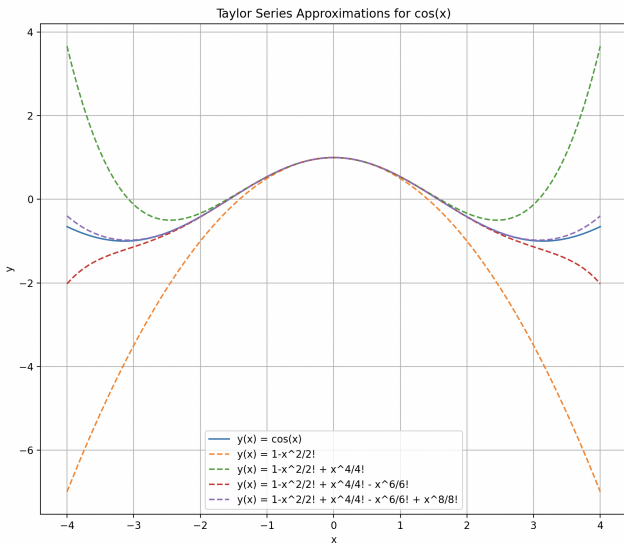
$$\cos(x) = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!} x^{2n}. \quad (17)$$

Hence, $C_n = (-1)^n/(2n)!$ and:

$$\lim_{n \rightarrow \infty} \left[\frac{|C_n(a)|}{|C_{n+1}(a)|} \right] = \lim_{n \rightarrow \infty} \left[\frac{\frac{(-1)^n}{2n!}}{\frac{(-1)^{(n+1)}}{(2n+2)!}} \right] = \lim_{n \rightarrow \infty} (2n+1)(2n+2) = \infty. \quad (18)$$

The series converges for all real x .

Example 2: Taylor Series Expansion for $\cos(x)$



Summary: Taylor Series Expansion

Strengths:

- When they are used to approximate complex functions with polynomials, the latter are **much easier** to **differentiate** and **integrate**.
- Can be easily manipulated to provide **finite difference approximations** to **function derivatives**.
- Can be used to get **theoretical error bounds**.

Weaknesses:

- Taylor series approximations **require underlying function** to be **continuously differentiable**.
- Series **convergence** for a **wide range of values** may only occur with an **unreasonably large** number of terms, a task that may be **computationally infeasible**.

Fourier Series and Fourier Integral

Fourier Series (Motivating Idea)

Fourier Series

A Fourier series is an expansion of a **periodic function** $f(x)$ in terms of an **infinite sum** of **trigonometric** (i.e., sines and cosines) and/or **exponential functions**.

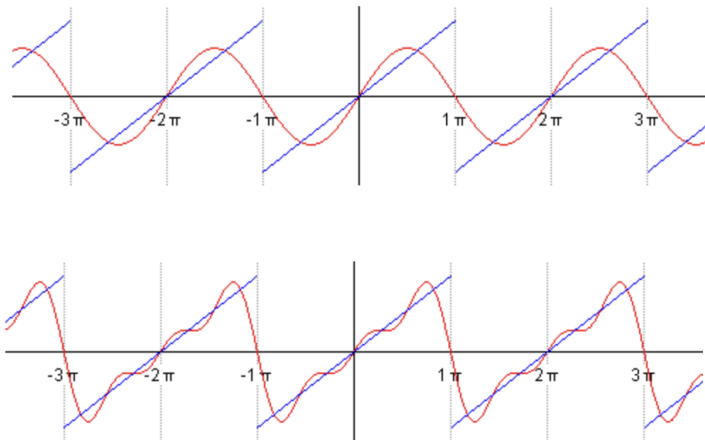
Periodic Function

A function $f(x)$ is **periodic** if and only if there exists a **positive number** $2p$ such that for every x in the domain of f , $f(x + 2p) = f(x)$. The number $2p$ is called the **period of f** .

Applications: Modeling of waveforms (e.g., surface waves on ocean; ocean tides; acoustics, musical tones; weather phenomena), analysis of resonant frequencies in a structure.

Fourier Series

Example 1: Progressive refinement of sawtooth function:



Fourier Series (Mathematics)

Mathematics: (trigonometry version) For $x \in [0, 2P]$:

$$f(x) \approx A_o + \sum_{n=1}^{\infty} \left[A_n \cos\left(\frac{n\pi x}{P}\right) + B_n \sin\left(\frac{n\pi x}{P}\right) \right]. \quad (19)$$

The coefficients are given by:

$$A_o = \frac{1}{P} \int_0^{2P} f(x) dx, \quad (20)$$

and
$$A_n = \frac{1}{P} \int_0^{2P} f(x) \cos\left(\frac{n\pi x}{L}\right) dx, \quad \text{for } n \geq 1, \quad (21)$$

$$B_n = \frac{1}{P} \int_0^{2P} f(x) \sin\left(\frac{n\pi x}{L}\right) dx, \quad \text{for } n \geq 1. \quad (22)$$

Fourier Series (Mathematics)

Example 1: Fourier Series for Sawtooth Function ...

The sawtooth shape can be written:

$$f(x) = \begin{cases} x & 0.0 \leq x \leq \pi, \\ x - 2\pi & \pi < x \leq 2\pi \end{cases} \quad (23)$$

Substituting 23 into 20 – 22 gives:

$$A_n = \frac{1}{\pi} \int_0^{2\pi} f(x) \cos(nx) dx = 0.0. \quad (24)$$

$$B_n = \frac{1}{\pi} \int_0^{2\pi} f(x) \sin(nx) dx = -\frac{2}{n} (-1)^n. \quad (25)$$

Fourier Series (Mathematics)

Example 1: Sawtooth function continued ...

The Fourier expansion is:

$$\begin{aligned} f(x) &= 2 \left[\sin(x) - \frac{\sin(2x)}{2} + \frac{\sin(3x)}{3} - \dots + (-1)^{n+1} \frac{\sin(nx)}{n} \dots \right] \\ &= 2 \sum (-1)^{n+1} \frac{\sin(nx)}{n}, \quad \text{for } -\pi \leq x \leq \pi. \end{aligned} \tag{26}$$

Note 1: The Fourier series covers only conditionally because of the discontinuity of $f(x)$ at $x \pm \pi$.

Note 2: At the discontinuity itself, the Fourier series will cover to the arithmetic mean of the end values.

Fourier Series (Mathematics)

Mathematics: (exponential series). Recall Euler's formula:

$$e^{inx} = \cos(nx) + i \sin(nx). \quad (27)$$

Rearranging equation 27 gives:

$$\cos(nx) = \left[\frac{e^{inx} + e^{-inx}}{2} \right], \quad \sin(nx) = \left[\frac{e^{inx} - e^{-inx}}{2i} \right]. \quad (28)$$

Substituting 27 into 19, and rearranging terms gives:

$$f(x) = \frac{A_0}{2} + \sum_{n=1}^{\infty} \left[A_n \left[\frac{e^{inx} + e^{-inx}}{2} \right] + B_n \left[\frac{e^{inx} - e^{-inx}}{2i} \right] \right]. \quad (29)$$

Fourier Series

Simplifying:

$$f(x) = \sum_{n=-\infty}^{\infty} c_n e^{-i\pi nx/L} \quad (30)$$

where:

$$c_n = \frac{1}{2} [A_n - iB_n], c_{-n} = \frac{1}{2} [A_n + iB_n], c_0 = \frac{A_0}{2}. \quad (31)$$

Fourier Integral Analysis

Motivation:

- In many **practical problems**, the function involved is **non-periodic** (i.e., Fourier series are not possible).

Solution:

- Consider limiting form of Fourier Series when $p \rightarrow \infty$.
- Fourier Series becomes Fourier Integral Analysis

Fourier Transform Pair: Waveforms in time/frequency domains:

$$\begin{aligned} f(t) &= \int_{-\infty}^{\infty} g(w) e^{iwt} dw. \\ g(w) &= \frac{1}{2\pi} \int_{-\infty}^{\infty} f(\tau) e^{i w \tau} d\tau. \end{aligned} \tag{32}$$

Discrete Fourier Transform

Discrete Fourier Transform (DFT)

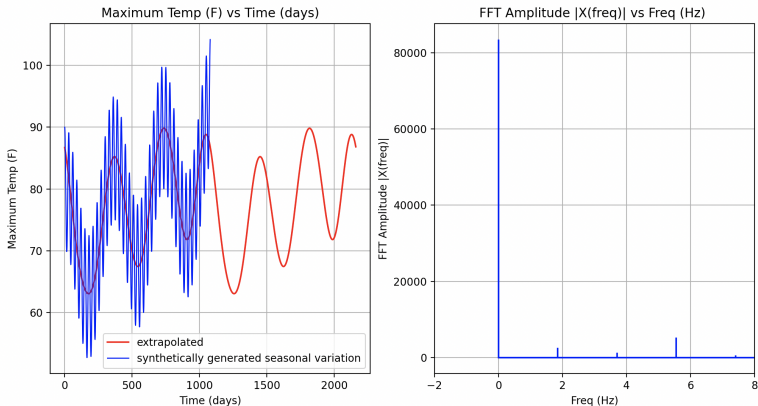
- Represents waveforms in both the time and frequency domains.
- Standard implementation of DFT requires $O(n^2)$ computational work, where n is the size of the data.

Fast Fourier Transform (FFT)

- An algorithm that computes the DFT of a sequence, most often in the time domain to a representation in the frequency domain.
- The inverse FFT transforms the frequency domain representation back into the time domain.
- FFT requires $O(n \log(n))$ computational work, so is very efficient.

Fast Fourier Transform Analysis

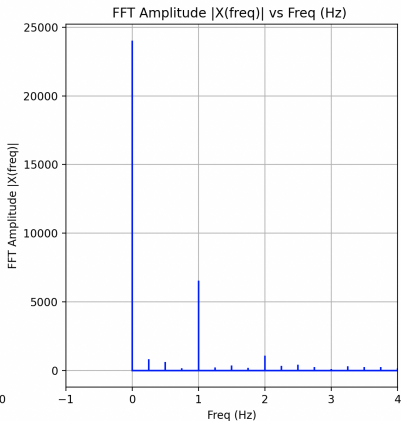
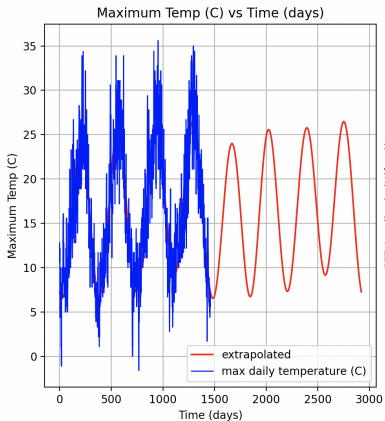
Example 1: FFT analysis of synthetically generated time series ...



Source Code: See python-code.d/math/

Fast Fourier Transform Analysis

Example 2: FFT analysis of max daily temperature in Seattle.



Source Code: See python-code.d/math/

Extra Slides

Extra Slides

Mean Value Theorem

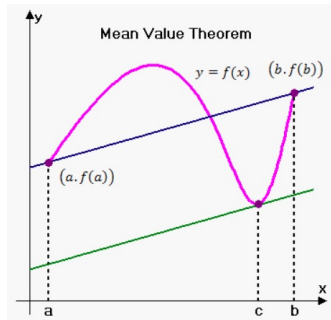
Let function f be continuous on $[a, b]$
and differentiable on (a, b) .

There exists a point c in (a, b) such that:

$$f'(c) = \left[\frac{f(b) - f(a)}{b - a} \right]. \quad (33)$$

The right-hand side of the equation is
the secant line connecting end points.

The green line is the tangent slope at
point c .



Python Code Listings

Code 1: Taylor Series Approximations for $\cos(x)$

```
1 # =====
2 # TestTaylorSeries02.py: Compute and plot Taylor Series approximations for
3 # cos(x).
4 #
5 # Written By: Mark Austin                                     August 2023
6 # =====
7
8 import math;
9 import numpy as np;
10 import matplotlib.pyplot as plt;
11
12 # Taylor series approximation for cos(x) ...
13
14 def func_cos(x,n):
15     cos_approx = 0
16     for i in range(n):
17         coeff = (-1)**i
18         num   = x**(2*i)
19         denom = math.factorial(2*i)
20         cos_approx = cos_approx + (coeff)*(num/denom)
21
22     return cos_approx
23
24 # main method ...
25
26 def main():
27     print("--- Enter TestTaylorSeries02.main()           ... ");
28     print("--- =====                               ... ");
```

Code 1: Taylor Series Approximations for $\cos(x)$

```

29
30     # Part 1: Compute x values for numerical computation ...
31
32     x = np.linspace( -4.0, 4.0, num = 101)
33     y = np.cos(x)
34
35     print(x)
36     print(y)
37
38     # Part 2: Two-term polynomial approximation:  $y(x) = 1 - x^2/2!$  ...
39
40     yapprox2 = []
41     for xi in x:
42         yapprox2.append( func_cos(xi,2) );
43
44     # Part 3: Three-term polynomial approximation:  $y(x) = 1 - x^2/2! + x^4/4!$  ...
45
46     yapprox3 = []
47     for xi in x:
48         yapprox3.append( func_cos(xi,3) );
49
50     # Part 4: Four-term polynomial approximation:  $y(x) = 1 - x^2/2! + x^4/4! - x^6/6!$  ..
51
52     yapprox4 = []
53     for xi in x:
54         yapprox4.append( func_cos(xi,4) );
55
56     # Part 5: Five-term polynomial approximation:  $y(x) = 1 - x^2/2! + x^4/4! - x^6/6! +$ 
57
58     yapprox5 = []

```

Code 1: Taylor Series Approximations for $\cos(x)$

```

59     for xi in x:
60         yapprox5.append( func_cos(xi,5) );
61
62     # Part 6: Plot cos(x) and various polynomial approximations ...
63
64     plt.plot(x,          y, label="y(x) = cos(x)",          linestyle="--")
65     plt.plot(x, yapprox2, label="y(x) = 1-x^2/2!",          linestyle="--")
66     plt.plot(x, yapprox3, label="y(x) = 1-x^2/2! + x^4/4!", linestyle="--")
67     plt.plot(x, yapprox4, label="y(x) = 1-x^2/2! + x^4/4! - x^6/6!", linestyle="--")
68     plt.plot(x, yapprox5, label="y(x) = 1-x^2/2! + x^4/4! - x^6/6! + x^8/8!", linestyle="--")
69     plt.title("Taylor Series Approximations for cos(x)")
70     plt.xlabel('x')
71     plt.ylabel('y')
72     plt.grid()
73     plt.legend()
74     plt.show()
75
76     print("--- ===== ... ");
77     print("--- Leave TestTaylorSeries02.main() ... ");
78
79     # call the main method ...
80
81     main()

```

Code 2: FFT of Max Temperature in Seattle

```
1 # =====
2 # TestFastFourierTransform03.py: Compute FFT extrapolation and dominant
3 # frequencies in Temperature Measurements, Seattle, WA.
4 #
5 # Modified by: Mark Austin                               September 2023
6 # =====
7
8 from math import cos, pi
9 import numpy as np
10 import pandas as pd
11 import matplotlib.pyplot as plt
12
13 # =====
14 # main method ...
15 # =====
16
17 def main():
18     print("--- Enter TestFastFourierTransform03.main()      ...");
19     print("--- ===== ...");
20
21     # Part 1: Set parameters ...
22
23     print("--- Part 1: Set parameters ...");
24
25     sr = 365 # <-- sampling rate (no samples per year) ...
26
27     # Part 2: Read Seattle weather ...
```


Code 2: FFT of Max Temperature in Seattle

```
29     print("--- Part 2: Read datafile for weather in Seattle ...");
30
31     rainfall = pd.read_csv('../data/seattle-weather.csv')
32     x = np.array( rainfall['temp_max'].values)
33
34     # Part 3: Compute fast fourier transform for maximum temperature ...
35
36     print("--- Part 3: Compute FFT for max daily temperature in Seattle ...");
37
38     X = np.fft.fft(x)
39     N = len(X)
40     n = np.arange(N)
41     T = N/sr
42     freq = n/T
43
44     print("---           No samples per year = {:10.2f} ...".format(sr) );
45     print("---           No data points: N = {:10.2f} ...".format(N) );
46     print("---           No periods:      T = {:10.2f} ...".format(T) );
47     print(n)
48     print("--- Fourier transform result ...");
49
50     # print(X)
51
52     # Part 4: Plot data + extrapolated curve ...
53
54     print("--- Part 4: Plot data + extrapolated curve ...");
55
56     plt.figure(figsize = (12, 6))
57     plt.subplot(121)
```

Code 2: FFT of Max Temperature in Seattle

```
59     for num_ in [6]:
60         fft_list                = np.copy(X)
61         fft_list[num_:-num_] = 0
62
63         # Inverse Fast Fourier transform
64
65         t = np.fft.ifft(fft_list)
66
67         # Plot general trend ...
68
69         plt.plot(np.concatenate([t,t]), color = 'red', label='extrapolated' )
70
71     plt.plot( np.arange(0, x.size), x, 'b', label = 'max daily temperature (C)', linewidth
72     plt.title("Maximum Temp (C) vs Time (days)")
73     plt.ylabel("Maximum Temp (C)", fontsize=10, rotation=90)
74     plt.xlabel("Time (days)", fontsize=10, rotation=0)
75     plt.grid()
76     plt.legend()
77
78     # Part 5: Plot data in frequency domain ...
79
80     print("--- Part 5: Plot data in frequency domain ...");
81
82     plt.subplot(122)
83     plt.stem( freq, np.abs(X), 'b', markerfmt=" ", basefmt="-b")
84     plt.title("FFT Amplitude |X(freq)| vs Freq (Hz) ")
85     plt.xlabel('Freq (Hz)')
```

Code 2: FFT of Max Temperature in Seattle

```
86     plt.ylabel('FFT Amplitude |X(freq)|')
87     plt.grid()
88     plt.xlim(-1, 4)
89
90     plt.show()
91
92     print("--- ===== ... ");
93     print("--- Leave TestFastFourierTransform03.main() ...");
94
95     # call the main method ...
96
97     main()
```