Endsem Exam

On ...

PH1050 Computational Physics

Aarya Gosar
Engineering Physics
24th Nov 2023

Part -1

This is a Linear recursion series, a modified version of the Fibonacci sequence.
We can use Matrix exponentiation to compute for Large values of n
We have a base matrix for initial values
B = {2,1,0}
and a Transition Matrix
T = {{1,1,0},{1,0,1},{1,0,0}]
Each Time we multiply T, We get the next triplet.
This works because first term is linear addition of all 3 values. Second and Third Terms is just previously calculated.
Our final answer would be
B . T^(n-2)
To calculated exponent efficiently, We take those powers of T which are powers of 2
T, T^2, T^4 , T^8 T^16 ....
and write n in binary form. Eg 9 = 1001
and we multiply only those powers of T where there is a one,
Eg for n = 9 we multiply T^8 and T^1

This reduces the time complexity to O(Log(n)) rather than O(n)

In[544]:=

```
Clear["Global`*"]
Fibo[f0_ : 2,f1_ : 1,f2_ : 0 , t_] :=(
B = {f0,f1,f2};
T = {{1,1,0}, {1,0,1} , {1,0,0}};
n = t-2;
While[n ≥ 1,
If[Mod[n,2] == 1,
B = B.T;
] ;
T = T. T;
n = Floor[n/2];
];
Return[B⟦1⟧])

Print["Computing First 10 Terms: "]
first10 = {0,1,2};
For[i = 3,i≤ 13,i++,
AppendTo[first10, Fibo[2,1,0,i]]]
first10

Print["Calcaulating the time required for the given set"]
set = {1,2,4,8,10,20,40,80,100,200,400,800,1000, 2000, 4000, 8000 }
Timing[
Fibo[#]&/@set;]

Print["The code is so efficient that time is negligibl, hence testing on a bigger set"]
set2 = {10};
Do[AppendTo[set2,set2⟦-1⟧ * 10] , 7];
set2
Timing[
Fibo[#]&/@set2;]

Print["Plotting Time vs n on a ListLog Plot"]
times = {}
set3 = {1000};
Do[AppendTo[set3, set3⟦-1⟧ * 2], 18]
set3
For[i = 1, i ≤ 19,i++,
AppendTo[times, {set3⟦i⟧,Timing[Fibo[set3⟦i⟧]]⟦1⟧}]
]
ListLogPlot[times]
Print["The computational time also increases rapidly in this method because the computer has to p
```

Computing First 10 Terms:

Out[549]=

```
{0, 1, 2, 3, 6, 11, 20, 37, 68, 125, 230, 423, 778, 1431}
```

Calcaulating the time required for the given set

Out[551]=

```
{1, 2, 4, 8, 10, 20, 40, 80, 100, 200, 400, 800, 1000, 2000, 4000, 8000}
```

Out[552]=

```
{0., Null}
```

The code is so efficient that time is negligibl, hence testing on a bigger set

Out[556]=

```
{10, 100, 1000, 10 000, 100 000, 1 000 000, 10 000 000, 100 000 000}
```

Out[557]=

```
{5., Null}
```

Plotting Time vs n on a ListLog Plot

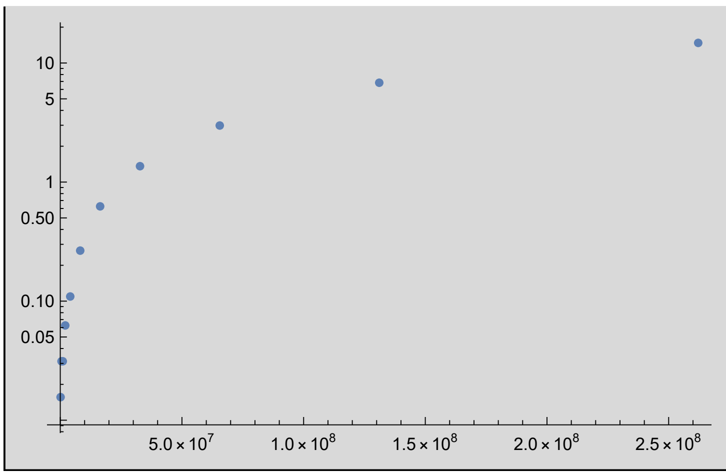
Out[559]=

```
{}
```

Out[562]=

```
{1000, 2000, 4000, 8000, 16 000, 32 000, 64 000, 128 000, 256 000, 512 000, 1 024 000,
 2 048 000, 4 096 000, 8 192 000, 16 384 000, 32 768 000, 65 536 000, 131 072 000, 262 144 000}
```

Out[564]=



The computational time also increases rapidly in this method because the computer has to process large numbers. but the number of steps of algorithm are still small

This method was taught to me by programming club of IITM

Part -2

In[587]:=

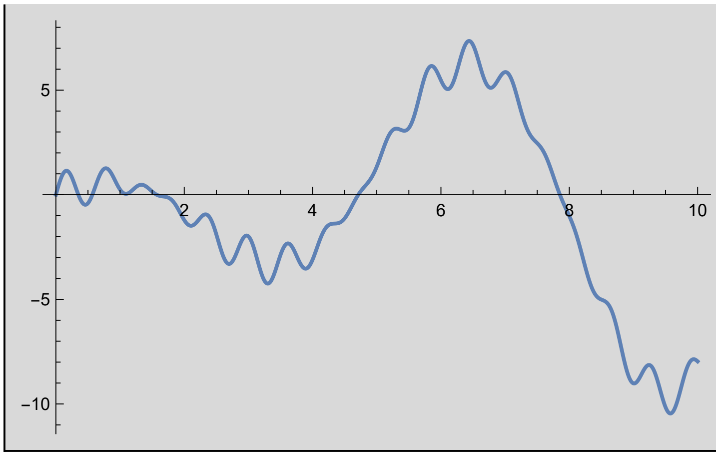
```
Clear["Global`*"]
Print["The function I have chosen is this"]
f[x_] = (x + Sin[10 x]) Cos[x]
ListLinePlot[Table[{x,f[x]} , {x,0,10,0.001}]]
Print["Gn is simply f'[x] written in ugly way"]
Print["Printing 60 values of Gn"]
Gn[h_ : 1/100, x_] := (2f[x+h] + 3 f[x] - 6 f[x-h] + f[x- 2 h])/ (6 h)
values = {};
v2 = {};
For[i = 1, i ≤ 60, i ++ ,
AppendTo[values, Gn[1 + 2 i]];
AppendTo[v2, f'[1 + 2i]]
]
values // N
Print["Printing 60 values of f'(x)"]
v2 // N
Print["The values are similar, this concludes that G must be d/dx operation"]
Print[" Plotting G - Gn for h going from 10^-1 to 10^-10"]
Print["On a Log-Log graph"]
e[h_] :=  Abs[f'[5] - Gn[h,5]]
table = {};
For[i = 1 , i ≤ 10 , i ++,
AppendTo[table, {(1/10)^(i), e[(1/10)^(i)]}]
]
ListLogLogPlot[table]
Print["On log log plot, the error grows linearly with h"]
```

The function I have chosen is this

Out[589]=

Cos[x] (x + Sin[10 x])

Out[590]=



Gn is simply f'[x] written in ugly way

Printing 60 values of Gn

Out[597]=

{-2.80013, 7.56355, 0.421814, -0.906884, 10.9162, -7.49769, -15.3606, 13.8205, -1.35206, -13.668, 22.6065, 6.55936, -28.8213, 14.8381, 8.97889, -32.7467, 16.2361, 29.9721, -35.3241, 5.89266, 31.5276, -41.0201, -9.91601, 49.944, -30.4089, -16.7242, 54.5737, -25.3143, -44.2466, 56.9384, -10.8119, -49.2017, 59.6181, 12.8623, -70.9239, 46.2815, 24.1471, -76.3946, 34.7368, 58.1759, -78.655, 16.1016, 66.6899, -78.4004, -15.3908, 91.7531, -62.4466, -31.2514, 98.207, -44.508, -71.7521, 100.466, -21.7538, -83.991, 97.3671, 17.4944, -112.424, 78.8951, 38.0407, -120.008}

Printing 60 values of f'(x)

Out[599]=

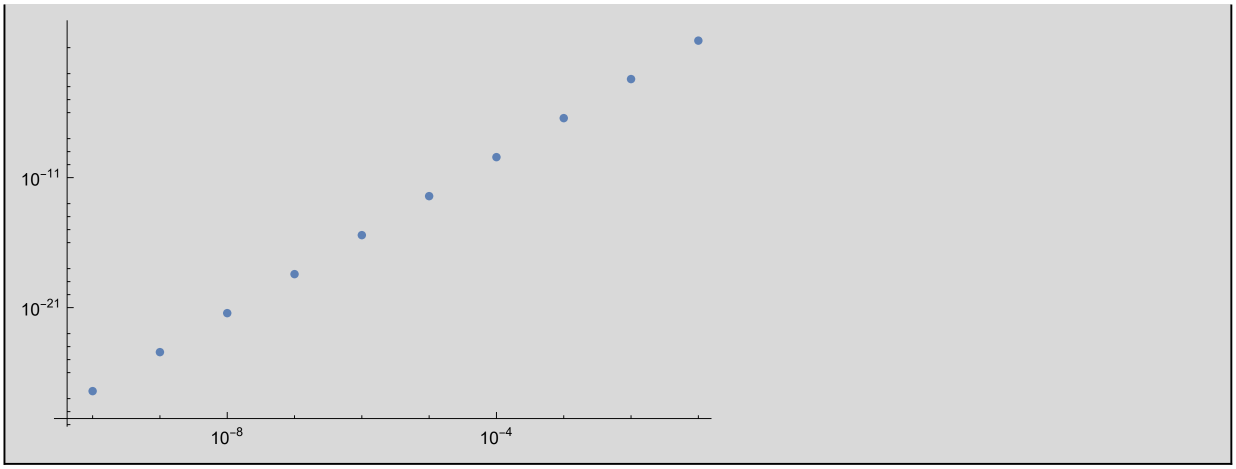
{-2.801, 7.56393, 0.421168, -0.906095, 10.9159, -7.4969, -15.3613, 13.8208, -1.35293, -13.6676, 22.606, 6.56023, -28.8217, 14.8388, 8.97812, -32.7464, 16.2353, 29.9728, -35.3245, 5.89354, 31.5271, -41.0196, -9.91685, 49.9444, -30.4096, -16.7235, 54.5734, -25.3135, -44.2472, 56.9388, -10.8127, -49.2013, 59.6175, 12.8631, -70.9242, 46.2822, 24.1464, -76.3943, 34.736, 58.1764, -78.6554, 16.1025, 66.6895, -78.3999, -15.3916, 91.7534, -62.4473, -31.2507, 98.2067, -44.5072, -71.7526, 100.466, -21.7547, -83.9906, 97.3666, 17.4951, -112.425, 78.8958, 38.0401, -120.008}

The values are similar, this concludes that G must be d/dx operation

Plotting G - Gn for h going from 10^-1 to 10^-10

On a Log-Log graph

Out[606]=



On log log plot, the error grows linearly with h

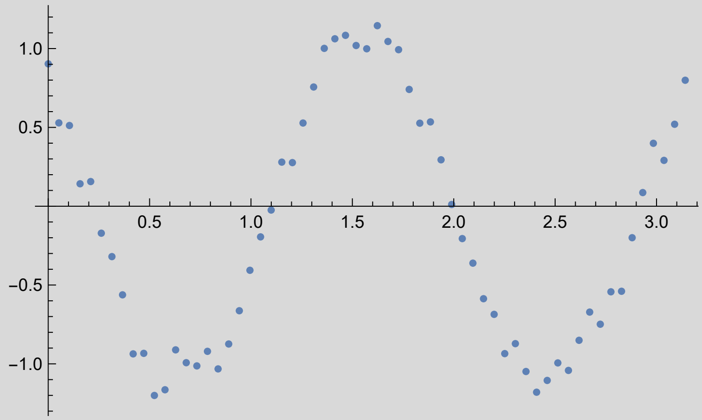
Part -3

In[608]:=

```
Clear["Global`*"]
Print["The data looks like sinosuidal plot"]
Data =Import["D:\\Aarya\\IITM\\First year\\data.csv"];
ListPlot[Data]
Print["Using a Non-Linear Fit for solving A Cos wt + B wt"]
NonlinearModelFit[Data, A Cos[w t] + B Sin[w t], {A,B,w} , t]
F[t_] = 0.7426113181227931` Cos[3.4850734151197487` t]-0.8383857929473916` Sin[3.485073415119748
Show[ListLinePlot[Table[{t,F[t]} , {t,0,π, 0.001}]],ListPlot[Data]]
w = 3.4850734151197487`
Print["a = "]
a =- F[0]- F'[0]
Print["b = "]
b = F[π] + F'[π]
Print["λ = "]
l = w^2
eqns = {y''[t] + l y[t] == 0, y[0] + y'[0] == - a, y[π] + y'[π] == b}
soln1 = NDSolveValue[eqns,y , {t,0,100}];
Print["The numerically solved function is"]
Show[ListLinePlot[Table[{t,soln1[t]} , {t,0,π, 0.001}]],ListPlot[Data]]
Print["subtracting k = "]
k =100111111  10^-23
a2 = a - k;
b2 = b-k;
l2 = l - k;
eqns2 = {y''[t] + l2 y[t] == 0, y[0] + y'[0] == - a2, y[π] + y'[π] == b2};
soln2 = NDSolveValue[eqns2,y , {t,0,100}];
Print["Plotting the difference between the two functions"]
ListLinePlot[Table[{t,Abs[soln1[t]-soln2[t]]} , {t,0,99,0.001}]]
Print["By subtracting two waves of different frequencies, we have created beats. the graph looks
```

The data looks like sinosuidal plot

Out[611]=



Using a Non-Linear Fit for solving A Cos wt + B wt

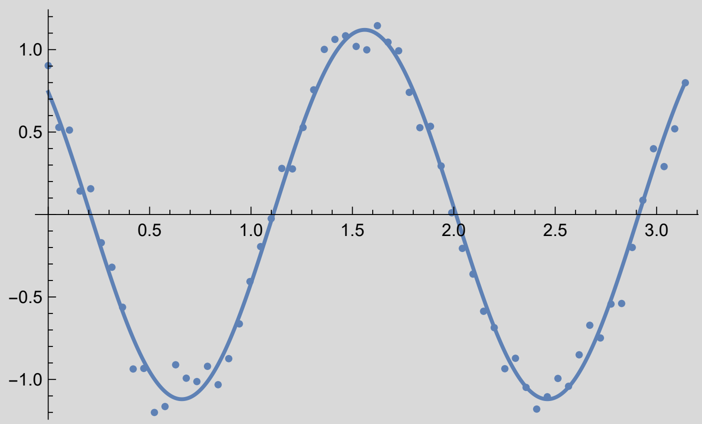
Out[613]=

```
FittedModel[ 0.742611 Cos[3.48507 t] – 0.838386 Sin[3.48507 t] ]
```

Out[614]=

```
0.742611 Cos[3.48507 t] - 0.838386 Sin[3.48507 t]
```

Out[615]=



Out[616]=

```
3.48507
```

a =

Out[618]=

```
2.17922
```

b =

Out[620]=

```
3.52483
```

λ =

Out[622]=

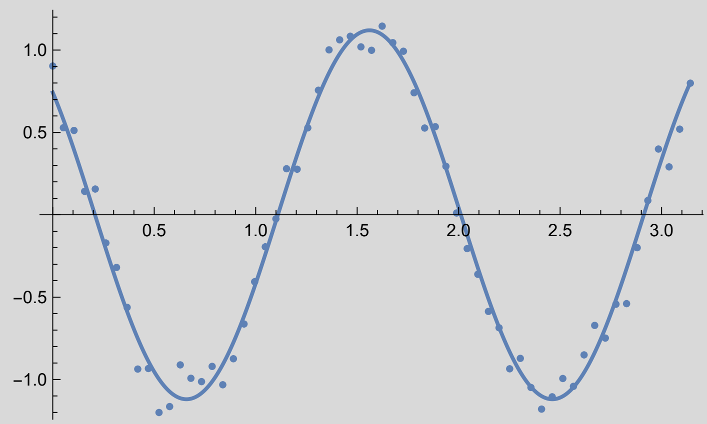
12.1457

Out[623]=

$\{12.1457\, y[t] + y''[t] == 0,\ y[0] + y'[0] == -2.17922,\ y[\pi] + y'[\pi] == 3.52483\}$

The numerically solved function is

Out[626]=



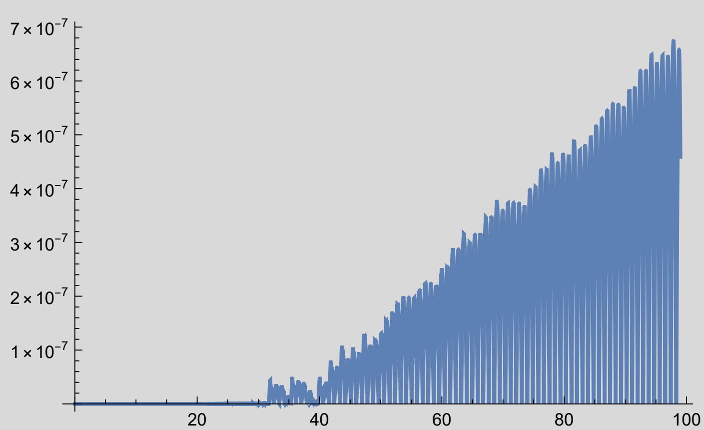
subtracting k =

Out[628]=

$$\frac{100\,111\,111}{100\,000\,000\,000\,000\,000\,000\,000}$$

Plotting the difference between the two functions

Out[635]=



By subtracting two waves of different frequencies, we have created beats. the graph looks linear because the beat freaquency is very low and we are looking at a small part