CS 3200 Assignment01

Lianrui Geng

U1346008

1. Question one is required to draw the plot without lines. The method about how to draw the plot is provided in the pdf which is fit the golden ratio. In my code, it defines a variable ratio equal to the value of (1+sqrt(5))/2, which is a mathematical constant known as the golden ratio.

The variable n is a vector with 4 elements [100 500 800 1100].

The code then enters a for-loop that runs 4 times, where i takes on the values 1 through 4.

For each iteration of the loop:

1. The variable current is defined as a range of numbers from 1 to the corresponding element of n at the current iteration (i.e. n(i)).
2. Two new variables r and theta are defined by performing mathematical operations on current. The r variable is the result of raising each element of current minus 1 to the power of the ratio constant. The theta variable is the result of multiplying 2 by each element of current minus 1, and then multiplying the result by the constant ratio and pi.
3. The subplot function creates a subplot with 2 rows and 2 columns, and the current iteration's plot (i) is placed in the subplot.
4. The polar plot function plots the theta and r values as polar coordinates in the subplot, with a marker style of "m".

After the four-loop, 4 polar plots will be displayed, each in a separate subplot of a 2x2 grid.

Here is the graph of Question one.

Chart, bubble chart

Description automatically generated

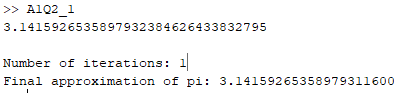
Question 2:

For the first part question, I think we need to find the count of terms when we used the formula to find the exact pi in the matlab.

This code calculates an estimate of the value of π (Pi), a well-known mathematical constant, using Chudnovsky's Algorithm.

The code starts by defining the variable n to be equal to 100 and two other variables, pi\_approx and pi\_approx\_sum, to be approximations of 0 with 100 digits of precision.

Then, the code enters a loop that runs n+1 times. Each time the loop runs, it calculates a new estimate of π using Chudnovsky's Algorithm. It does this by first calculating a term of the formula, then updating the value of pi\_approx\_sum by adding the current term to it. Next, the code calculates an updated estimate of π by dividing 1 by pi\_approx\_sum multiplied by 12.

The loop continues to run until the difference between the current estimate of π and the true value of π is small enough. At this point, the code prints the number of iterations it took to reach the final estimate, and the final estimate of π.

For the second part of this question, this code calculates an estimate of the value of π (Pi), a well-known mathematical constant, using Chudnovsky's Algorithm. The code calculates an estimate of π with increasing precision and measures the time it takes to do so.

The code starts by defining an array of digits, digits, with four elements: 10, 50, 100, and 500. Then, the code enters a loop that runs 4 times. Each time the loop runs, it calls the compute\_e function with a different element from the digits array as its argument. The compute\_e function returns an estimate of π with the specified number of digits of precision.

The compute\_e function starts by defining a symbolic variable k. It then calculates the terms of Chudnovsky's Algorithm using the symbolic variable k. The function uses the symsum function to sum up the terms from k = 0 to k = Inf. The function calculates the estimate of π by dividing 1 by 12 \* (y), where y is the sum of the terms. The function then returns the estimate of π with the specified number of digits of precision.

After calling the compute\_e function, the code measures the time it took to calculate the estimate of π using the toc function. The code then displays the estimate of π and the time it took to calculate it. The code repeats this process four times, with each iteration using a different number of digits of precision.

>> A1Q2\_2

3.141592654

1.8626

3.1415926535897932384626433832795028841971693993751

0.0352

3.141592653589793238462643383279502884197169399375105820974944592307816406286208998628034825342117068

0.0314

3.1415926535897932384626433832795028841971693993751058209749445923078164062862089986280348253421170679821480865132823066470938446095505822317253594081284811174502841027019385211055596446229489549303819644288109756659334461284756482337867831652712019091456485669234603486104543266482133936072602491412737245870066063155881748815209209628292540917153643678925903600113305305488204665213841469519415116094330572703657595919530921861173819326117931051185480744623799627495673518857527248912279381830119491

0.0512

Question 3: This code calculates and plots two graphs related to the velocity v of an object over time t. The velocity of the object is defined using a piecewise function v(t). The function defines the velocity of the object based on the range of t.

The first graph plots the velocity of the object as a function of time. This is done using the plot function, with the v function as the y-axis and the time values from 1 to 50 as the x-axis.

The second graph calculates the cumulative distance value traveled by the object over time and plots it. The cumulative distance is calculated using the trapz function, which integrates the velocity function over the time range from 1 to i, where i ranges from 2 to 50. The cumulative distance is then plotted as a function of time using the plot function, with the cumulative distance values as the y-axis and the time values from 1 to 50 as the x-axis.

Both graphs are displayed using the subplot function, with the first graph displayed in the first subplot and the second graph displayed in the second subplot.

