Documentation for Project Sinussum

Ferrulli Massimiliano

1 Outcome of the analysis phase

After creating a matrix filled with spaces, labeled as "empty_matrix", I fill the time axis with the char "." . This is possible with the function "assign_time_axe" that calls "time_index_i", this last function determines the row index i at which the time axis should be inserted into the matrix. Then, based on the signal input, "matrix_chosen" determines which theoretical values needs to be computed by calling one of the three theoretical functions (SIGNAL_theory[1]). These three functions are responsible for computing values for each cell along the temporal axis and assigning them to the input matrix using the '+' character. The same steps are done to calculated the approximated values but the assigned character is '*', the functions I use for this part are "functions_approx", "approximated_matrix_chosen", "value_SIGNAL_approx". The function that returns the row index i for every calculated value is "row_index". In order to print the graph on the terminal, every element of the matrix are printed row by row using "print", the horizontal bars are printed with "print_bars". For the dichotomic research of the maximum a function called "time_dichotomy" returns a vector containing in the first slot the t_start and in the second the t_finish, based on the signal input. The dichotomic research is made by using the iterative function "max_dicho_research" which selects, every iterations, in which side of the graph the research must continue.

2 Order of complexity of task 2

All "TYPE_theory" functions exhibit complexity $O(nbC) = O(nbL)$ $(nbC = 2 \cdot nbL - 1)$. "functions_approx" has complexity $O(nbL \cdot nbN)$ due to its for loop, which increments t from tmin to tmax with delta_t, requiring nbC iterations ($O(nbL)$). Within this loop, the "approximated_matrix_chosen" function is invoked with a complexity of $O(nbN)$ based on the sum of the fourier's formulas. The "print" function has a complexity $O(nbL^2)$ due to the two for loops, the final complexity is $O(nbL \cdot nbN) + O(nbL^2)$. The bigger terms between nbN and nbL will determinate which of the two sides of the sum will be the complexity.

3 Pseudocode for the dichotomic research of the maximum for the signal SQUARE (2nd Page)

4 Behaviour for $\lim\limits_{nbN \to \infty}$

While increasing nbN I notice that for the approximated maximum for the signal TRIANGLE tends to 1 while for SQUARE and SAWTOOTH the maximum tends to 1.17897 (value obtained with $nbN = 10^6$ and $nbN = 10^7$)

[1]The underlined word SIGNAL is used to not repeat the three signals, SQUARE,TRIANGLE and SAWTOOTH.
[1]For SQUARE 1.17897974, for SAWTOOTH 1.17897874

Algorithm 1: Max dicho research

Input: $t_{start}, t_{finish}, \varepsilon$ three decimal numbers, where: $t_{start} < t_{finish}$, $\varepsilon \in \mathbb{R}^{+}$ and $t_{start}, t_{finish} \geq 0$, an integer $nbN > 0$

Output: The maximum value approximated for the function SQUARE in the interval $\{t_{start}, t_{finish}\}$

$c \leftarrow 0$
while $|f(c) - f(c_{previous})| \geq \varepsilon$ **do**
  $c_{\text{previous}} \leftarrow c$
  $c \leftarrow \frac{t_{start}+t_{finish}}{2}$
  $f(t_{start}) \leftarrow \text{value\_square\_approx}(t_{start}, nbN)$
  $f(t_{finish}) \leftarrow \text{value\_square\_approx}(t_{finish}, nbN)$
  $f(c) \leftarrow \text{value\_square\_approx}(c, nbN)$
  $f(c_{\text{previous}}) \leftarrow \text{value\_square\_approx}(c_{\text{previous}}, nbN)$
  if $f(c) < f(t_{start})\, and\, f(c) > f(t_{finish})$ **then**
    $t_{finish} \leftarrow c$
  if $f(c) > f(t_{start})\, and\, f(c) < f(t_{finish})$ **then**
    $t_{start} \leftarrow c$
  if $(f(c) < f(t_{start})\, and\, f(c) < f(t_{finish}))$ *or* $(f(c) > f(t_{start})\, and\, f(c) > f(t_{finish}))$ **then**
    if $f(t_{finish}) < f(t_{start})$ **then**
      $t_{finish} \leftarrow c$
    if $f(t_{finish}) > f(t_{start})$ **then**
      $t_{start} \leftarrow c$
  if $(f(c) = f(t_{start}))$ *or* $(f(c) = f(t_{finish}))$ **then**
    if $f(t_{finish}) < f(t_{start})$ **then**
      $t_{finish} \leftarrow c$
    if $f(t_{finish}) > f(t_{start})$ **then**
      $t_{start} \leftarrow c$
return *f(c)*