Wrocław University of Science and Technology

Scientific Calculations - Task List 2

conditioning of tasks and stability of algorithms

Marcel Jerzyk
October 24, 2019

1 Impact of minor change in data on the calculation results

1.1 Problem

First experiment on this Task List was about investigating the results of calculating two scalar products of vectors when there was a tiny difference on the input data.

1.2 Results

Float32	
Old Frwd	-0.4999443
New Frwd	-0.4999443
Old Bkwr	-0.4543457
New Bkwr	-0.4543457
Old Desc	-0.25
New Desc	-0.25
Old Ascn	-0.25
New Ascn	-0.25

Table 1: No difference between Old and New results.

Float64	
Old Frwd	1.0251881368296672e-10
New Frwd	0.004296342739891585
Old Bkwr	-1.5643308870494366e-10
New Bkwr	-0.004296342998713953
Old Desc	0.0
New Desc	-0.004296342842280865
Old Ascn	0.0
New Ascn	-0.004296342842280865

Table 2: Huge difference in results.

The minor change is just about deleting 9 from x_4 and 7 from x_5. The results has changed (or not) as in *Table 1* and *Table 2*.

1.3 Conclusions

The change is not really about deleting just a random number from a random input, but rather deleting 10th digit in two inputs that had a value on that place. After deleting these two numbers all the inputs ends on 9th place after the digit.

This brings no change in the *Float32* arithmetic but in *Float64* we can clearly see a difference. This brings a conclusion that this is an ill-conditioned task where small change in input data causes big changes in the results.

2 Visualization of a function

2.1 Problem

$$f(x) = e^x * ln(1 + e^{-x}) \tag{1}$$

This task required drawing the above function with programs visualisation programs and calculating $f(x) = \lim_{x\to\infty} f(x)$.

2.2 Results

I used "*desmos.com*", "*wolframalpha.com*", "*graphsketch.com*" and "*fooplot.com*" which can be used through web to complete this task.

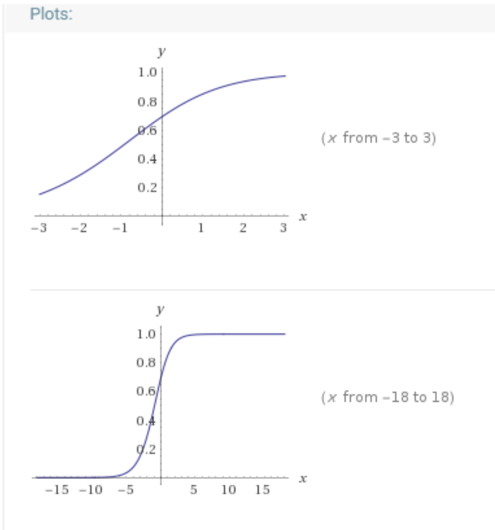


Figure 1: Wolframalpha.com

Figure 2: Desmos.com

2.3 Conclusions

The limit of the function at $x \to 0$ is equal to 1, but on the graphs that I was able to draw beyond x = 3 (on graphs exceeding x = 31 to be exact) the function starts to fluctuate at $x \geq 32$ which is not correct with the previously calculated limit. The error is caused by multiplying something very big (e^x) and something very small (logarithm). After $x = 36$ the function is equal to 0 which is because of the very small (e^{-x}) that after this point is equal to 0.

Such equation that makes the algorithm calculating and drawing this $f(x)$ is called numerically unstable because of dramatic results throughout wrongly calculated steps.

2.4 Close-up on the graph in the . miss behaving range

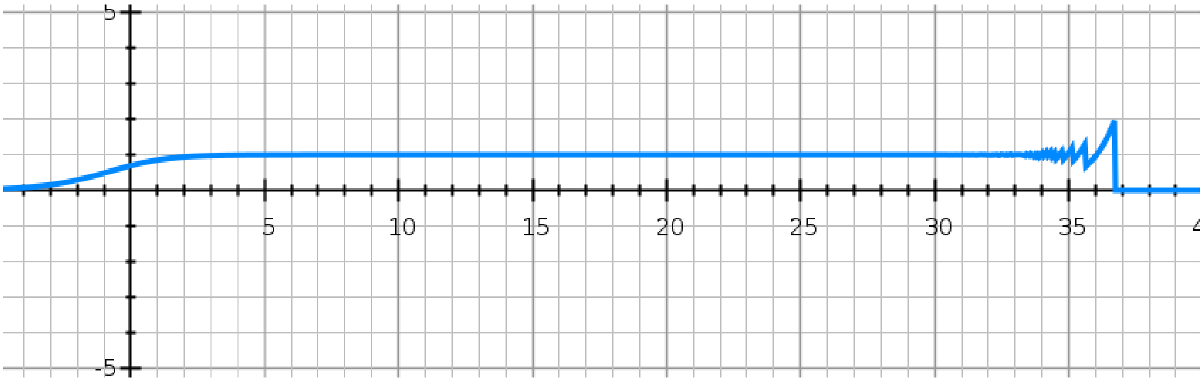


Figure 3: Graphsketch.com

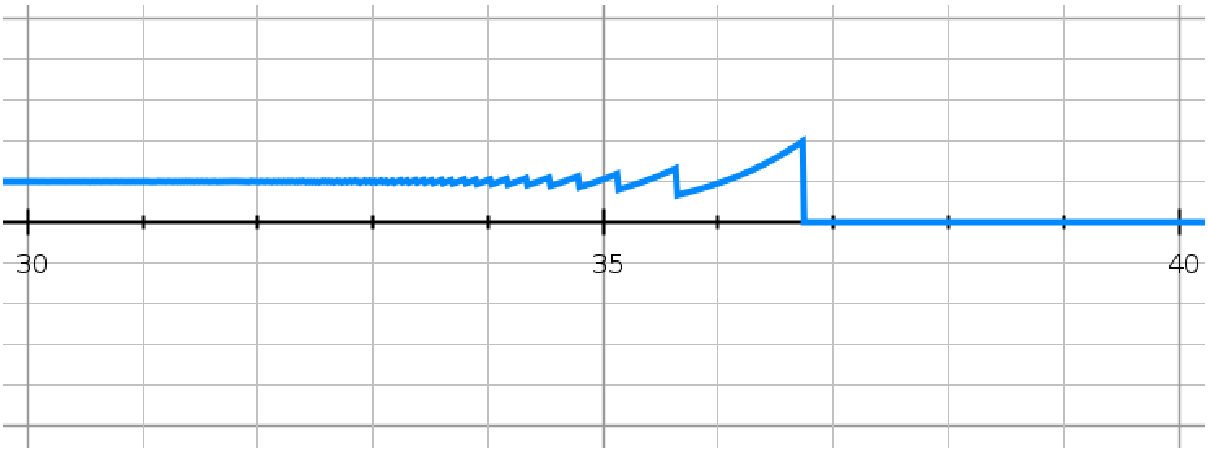


Figure 4: Graphsketch.com in range x = <30, 40>

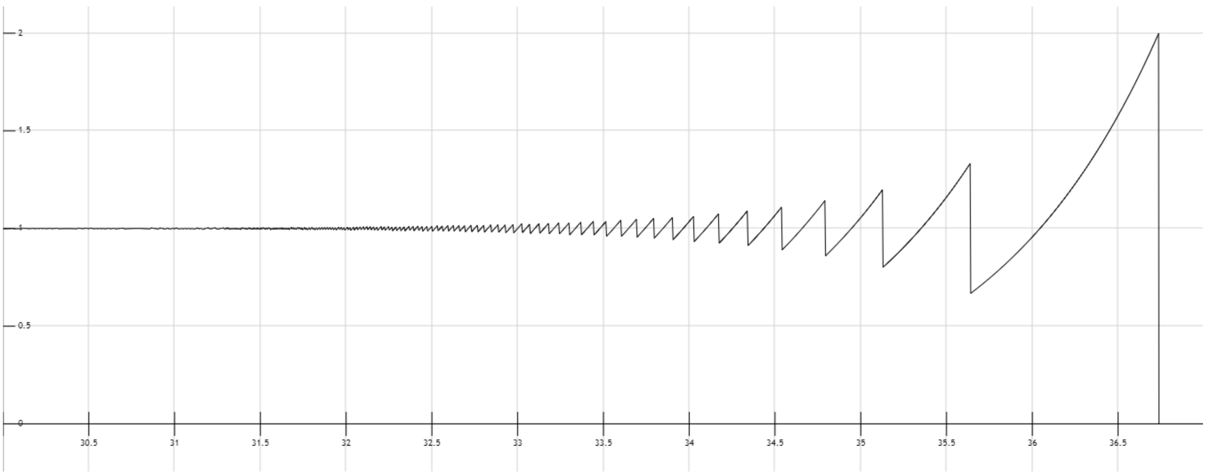


Figure 5: Fooplot.com in range x = <30, 37>

3 Solving linear equations with Gauss Elimination and inversion

3.1 Problem

This task is about simply solving linear equation

$$Ax = b \tag{2}$$

for a matrix with coefficients $A \in R^{n \times n}$ and vector $b \in R^n$. The matrix was genereted in two different ways:

- $A = H_n$ where H_n was a *Hilbert Matrix* of a degree of n generated by hilb(n).
- $A = R_n$ where R_n was a *random matrix* of a degree of n and conditioning indicator c generated by matcond(n, c).

I had to use use *Gauss Elimination* ($x = A \div b$) and *matrix inversion* ($x = A^{-1} * b$) to solve it for both versions of matrices. Hilberts with on growing degree of n and random with different conditioning indicators topping it with relative errors.

3.2 Results

The results are presented on the next page.

3.3 Conclusions

Cond() was used to calculate matrix conditioning indicator, which allows estimating accuracy to how many decimal places it is able to give the correct result.

The relative error on *Hilbert Matrices* increases with the increase of the conditioning indicator. The increase in relative error for both methods of solving is also directly related to the increase in the degree of the matrix. The same goes for the random matrices but not as significantly.

It's yet another ill conditioned task because with the increase of degrees, the cond increases rapidly and so is the relative error.

n	Gauss	Inversion	Cond
2	5.661048867003676e-16	1.4043333874306803e-15	19.28147006790397
3	8.022593772267726e-15	0.0	524.0567775860644
4	4.137409622430382e-14	0.0	15513.73873892924
5	1.6828426299227195e-12	3.3544360584359632e-12	476607.25024259434
6	2.618913302311624e-10	2.0163759404347654e-10	1.4951058642254665e7
7	1.2606867224171548e-8	4.713280397232037e-9	4.75367356583129e8
8	6.124089555723088e-8	3.07748390309622e-7	1.5257575538060041e10
9	3.8751634185032475e-6	4.541268303176643e-6	4.931537564468762e11
10	8.67039023709691e-5	0.0002501493411824886	1.6024416992541715e13
11	0.00015827808158590435	0.007618304284315809	5.222677939280335e14
12	0.13396208372085344	0.258994120804705	1.7514731907091464e16
13	0.11039701117868264	5.331275639426837	3.344143497338461e18
14	1.4554087127659643	8.71499275104814	6.200786263161444e17
15	4.696668350857427	7.344641453111494	3.674392953467974e17
16	54.15518954564602	29.84884207073541	7.865467778431645e17
17	13.707236683836307	10.516942378369349	1.263684342666052e18
18	9.134134521198485	7.575475905055309	2.2446309929189128e18
19	9.720589712655698	12.233761393757726	6.471953976541591e18
20	7.549915039472976	22.062697257870493	1.3553657908688225e18

Table 3: Relative Errors for Hilbert Matrices.

n	Gauss	Inversion	Cond
5	0.0	1.9860273225978183e-16	1.0000000000000007
5	1.4895204919483638e-16	2.220446049250313e-16	9.999999999999988
5	1.4143735885639737e-14	1.43554885227802e-14	1000.000000000027
5	1.2167207873035827e-11	3.4436116595168225e-11	9.999999987763042e6
5	2.6710906373707157e-5	2.290407265057079e-5	9.999497108053284e11
5	0.38309697881394983	0.3770775782249589	5.992545671809469e15
10	1.6837365821701475e-16	2.1642230995786354e-16	1.0000000000000013
10	4.1540741810552243e-16	4.2711132545550575e-16	10.0
10	4.58969071952651e-14	4.596205325715962e-14	1000.0000000000155
10	3.019938740923405e-10	3.4492950780370644e-10	1.0000000002362452e7
10	3.5664222293310965e-5	4.04216183710532e-5	9.999688872223783e11
10	0.02360686172731404	0.07690514942483787	5.822469574502952e15
20	5.639233568644062e-16	3.657001166779273e-16	1.0000000000000013
20	8.770059193588642e-16	9.942541535518373e-16	9.99999999999999
20	8.526773033672253e-15	1.2620652816928193e-14	999.9999999999768
20	3.331575258102934e-10	3.3976852381714325e-10	1.0000000001067158e7
20	9.078247865384858e-7	3.0966893227797827e-6	1.000006872275114e12
20	0.14004962504800678	0.08291930023560558	1.1259926292207776e16

Table 4: Relative Errors for Random Matrices.

4 Wilkinson Polynomial

4.1 Problem

The task is to calculate twenty zero places of the polynomial below. P is natural version of Wilkinson polynomial p.

$$\begin{aligned} P(x) \;=\; & x^{20} - 210x^{19} + 20615x^{18} - 1256850x^{17} + 53327946x^{16} \\ & -1672280820x^{15} + 40171771630x^{14} - 756111184500x^{13} \\ & +11310276995381x^{12} - 135585182899530x^{11} \\ & +1307535010540395x^{10} - 10142299865511450x^{9} \\ & +63030812099294896x^{8} - 311333643161390640x^{7} \\ & +1206647803780373360x^{6} - 3599979517947607200x^{5} \\ & +8037811822645051776x^{4} - 12870931245150988800x^{3} \\ & +13803759753640704000x^{2} - 8752948036761600000x \\ & +2432902008176640000 \end{aligned}$$

$$\begin{aligned} p(x) \;=\; & (x-20)(x-19)(x-18)(x-17)(x-16) \\ & (x-15)(x-14)(x-13)(x-12)(x-11) \\ & (x-10)(x-9)(x-8)(x-7)(x-6) \\ & (x-5)(x-4)(x-3)(x-2)(x-1) \end{aligned}$$

After that, I have to check these roots z_k, $1 \le k \le 20$, by calculating $|P(z_k)|$, $|p(z_k)|$ and $|z_k - k|$. Upon finishing - repeat but with different value for the factor at x^{19}.

$$-210x^{19} \rightarrow -210 - 2^{-23} \tag{3}$$

4.2 Results

The results are presented on the next page.

4.3 Conclusions

As value of the roots increases so does the error value. Accurate representation of the coefficients is not possible because there are only 15-17 places for the representation of significant digits of a number in the *Float64* arithmetic - hence the disturbances. Calculations made for the second polynomial lead to similar conclusions. However, the changed data highlights the differences between relative errors, causing them to increase relative to the previous polynomial. The roots obtained after this small change belong to the set of complex numbers. Therefore - ill conditioned task as small changes in input have significant impact on result. Reminder: we decreased one value by 2^{-23}. Really makes you think...

Regular Experiment				
k	z_k	$\|P(z_k)\|$	$\|p(z_k)\|$	$\|z_k - k\|$
1	0.9999999999996989	36352.0	36352.0	3.0109248427834245e-13
2	2.0000000000283182	181760.0	181760.0	2.8318236644508943e-11
3	2.9999999995920965	209408.0	209408.0	4.0790348876384996e-10
4	3.9999999837375317	3.106816e6	3.106816e6	1.626246826091915e-8
5	5.000000665769791	2.4114688e7	2.4114688e7	6.657697912970661e-7
6	5.999989245824773	1.20152064e8	1.20152064e8	1.0754175226779239e-5
7	7.000102002793008	4.80398336e8	4.80398336e8	0.00010200279300764947
8	7.999355829607762	1.682691072e9	1.682691072e9	0.0006441703922384079
9	9.002915294362053	4.465326592e9	4.465326592e9	0.002915294362052734
10	9.990413042481725	1.2707126784e10	1.2707126784e10	0.009586957518274986
11	11.025022932909318	3.5759895552e10	3.5759895552e10	0.025022932909317674
12	11.953283253846857	7.216771584e10	7.216771584e10	0.04671674615314281
13	13.07431403244734	2.15723629056e11	2.15723629056e11	0.07431403244734014
14	13.914755591802127	3.65383250944e11	3.65383250944e11	0.08524440819787316
15	15.075493799699476	6.13987753472e11	6.13987753472e11	0.07549379969947623
16	15.946286716607972	1.555027751936e12	1.555027751936e12	0.05371328339202819
17	17.025427146237412	3.777623778304e12	3.777623778304e12	0.025427146237412046
18	17.99092135271648	7.199554861056e12	7.199554861056e12	0.009078647283519814
19	19.00190981829944	1.0278376162816e13	1.0278376162816e13	0.0019098182994383706
20	19.999809291236637	2.7462952745472e13	2.7462952745472e13	0.00019070876336257925
Changed: $-210x^{19} \to -210 - 2^{-23}$				
k	z_k	$\|P(z_k)\|$	$\|p(z_k)\|$	$\|z_k - k\|$
1	0.9999999999998357 + 0.0im	20992.0	20992.0	1.6431300764452317e-13
2	2.0000000000550373 + 0.0im	349184.0	349184.0	5.503730804434781e-11
3	2.99999999660342 + 0.0im	2.221568e6	2.221568e6	3.3965799062229962e-9
4	4.000000089724362 + 0.0im	1.046784e7	1.046784e7	8.972436216225788e-8
5	4.99999857388791 + 0.0im	3.9463936e7	4.2535936e7	1.4261120897529622e-6
6	6.000020476673031 + 0.0im	1.29148416e8	2.04793344e8	2.0476673030955794e-5
7	6.99960207042242 + 0.0im	3.88123136e8	1.754868736e9	0.00039792957757978087
8	8.007772029099446 + 0.0im	1.072547328e9	1.852128e10	0.007772029099445632
9	8.915816367932559 + 0.0im	3.065575424e9	1.37168464896e11	0.0841836320674414
10	10.095455630535774 - 0.6449328236240688im	7.143113638035824e9	1.4912572850824043e12	0.6519586830380406
11	10.095455630535774 + 0.6449328236240688im	7.143113638035824e9	1.4912572850824043e12	1.1109180272716561
12	11.793890586174369 - 1.6524771364075785im	3.357756113171857e10	3.2960224849741504e13	1.665281290598479
13	11.793890586174369 + 1.6524771364075785im	3.357756113171857e10	3.2960224849741504e13	2.045820276678428
14	13.992406684487216 - 2.5188244257108443im	1.0612064533081976e11	9.545941965367332e14	2.5188358711909045
15	13.992406684487216 + 2.5188244257108443im	1.0612064533081976e11	9.545941965367332e14	2.7128805312847097
16	16.73074487979267 - 2.812624896721978im	3.315103475981763e11	2.7420894080997828e16	2.9060018735375106
17	16.73074487979267 + 2.812624896721978im	3.315103475981763e11	2.7420894080997828e16	2.825483521349608
18	19.5024423688181 - 1.940331978642903im	9.539424609817828e12	4.252502487879955e17	2.454021446312976
19	19.5024423688181 + 1.940331978642903im	9.539424609817828e12	4.252502487879955e17	2.004329444309949
20	20.84691021519479 + 0.0im	1.114453504512e13	1.3743733195398482e18	0.8469102151947894

Table 5: Both Wilkinson Polynomial Experiments Results.

5 Studying recursive function of population growth model

5.1 Problem

The task wanted me to perform two experiments using population growth model (recursive function below).

$$P_{n+1} = p_n + rp_n(1 - p_n) \quad (4)$$

In the function: $n = 0, 1, 2...$ and r is a certain constant. $r(1-p_n)$ is the population growth factor, and p_0 is the population size as a percentage of the maximum population size for a given state of the environment.

With $p_0 = 0.01$ and $r = 3$ I had to:

- make an experiment in *Float32* for 40 iterations.
- make an experiment in *Float64* for 40 iterations.
- make an experiment in *Float32* for 10 iterations, trunk the result after third digit, continue with new value for next 30 iterations.

5.2 Results

n	Float32 Cut	Float32	Float64
8	0.056273222	0.056273222	0.056271577646256565
9	0.21559286	0.21559286	0.21558683923263022
10	0.722	0.7229306	0.722914301179573
11	1.3241479	1.3238364	1.3238419441684408
12	0.036488414	0.037716985	0.03769529725473175
13	0.14195944	0.14660022	0.14651838271355924
14	0.50738037	0.521926	0.521670621435246
15	1.2572169	1.2704837	1.2702617739350768
16	0.28708452	0.2395482	0.24035217277824272
17	0.9010855	0.7860428	0.7881011902353041
18	1.1684768	1.2905813	1.2890943027903075
⋮			
25	1.0929108	1.0070806	1.315588346001072
26	0.7882812	0.9856885	0.07003529560277899
27	1.2889631	1.0280086	0.26542635452061003
28	0.17157483	0.9416294	0.8503519690601384
29	0.59798557	1.1065198	1.2321124623871897
30	1.3191822	0.7529209	0.37414648963928676
⋮			
35	0.034241438	1.021099	0.9253821285571046
36	0.13344833	0.95646656	1.1325322626697856
37	0.48036796	1.0813814	0.6822410727153098
38	1.2292118	0.81736827	1.3326056469620293
39	0.3839622	1.2652004	0.0029091569028512065
40	1.093568	0.25860548	0.011611238029748606

Table 6: Representation of all the experiments.

5.3 Conclusion

Single precision is not enough to get correct results. In subsequent loop iterations I noticed a growing difference between the values for the *Float32* and *Float64* arithmetic. This is because there are too few places available for significant numbers in *Float32* arithmetic. On the other hand, by performing 40 iterations in *Float32* without interruption, we get different results than if we would cut the result to 3 significant digits after 10th iteration. The absolute error for the initial value was 0.0 but after 10th iteration it slowly increased and caused a significant disturbance of the result. Another: small change in input - big difference in result.

Last result which is p_{40} in *Float64* is 109 times smaller than the p_{40} in *Float32* with trunk in one of the iterations.

The results are about the same until 15th iteration. After $n \geq 16$ the results from all the methods start to differ noticeably.

6 Studying recursive quadratic function

6.1 Problem

The task is about another recursive function - this time a quadratic one.

$$x_{n+1} = x_n^2 + c \tag{5}$$

Once again, I'm going to iterate through this function 40 times but this time only in arithmetic *Float64*. There are required inputs:

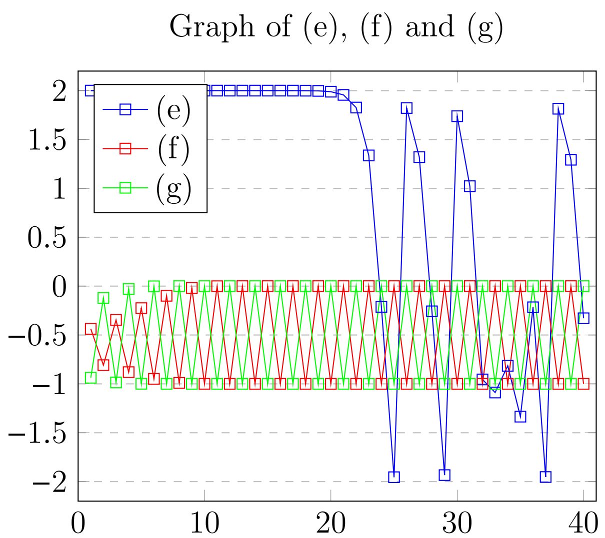
(a) $c = -2 \wedge x_0 = 1$
(b) $c = -2 \wedge x_0 = 2$
(c) $c = -1 \wedge x_0 = 1$
(d) $c = -1 \wedge x_0 = -1$
(e) $c = -1 \wedge x_0 = 0.75$
(f) $c = -1 \wedge x_0 = 0.25$
(g) $c = -2 \wedge x_0 = 1.99999999999999$

6.2 Results

Some of the inputs have regular results (*(a), (b), (c) and (d)*), others diverge (*(e), (f), (g)*). I showcase them accordingly to values.

C	X	Value Returned
-2	1	-1
-2	2	2
-1	1	[0, -1, 0, -1, ...]
-1	-1	[0, -1, 0, -1, ...]

Table 7: Regular results of some of the inputs.



6.3 Conclusion

Results from (e), (f) and (g) for given data sets represent deterministic chaos, where errors begin to accumulate and increase with each subsequent iteration. The error at the output is moved to the input of the next iteration.

On the other hand (a), (b), (c) and (d) show the opposite situation, where the obtained results are predictable and in line with expectations. Data instability does not result from the x^2 but from the data given when the function was called. If they are integers, then the graph begins to oscillate. However, when the data is more precise, such behavior occurs only after a greater number of iterations.