Assignment № 2

Wei Huang, Università della Svizzera italiana 23/04/2020

Problem 1

I wrote one python code which could implement even and odd method to approximate the first order derivative of the function

$$u(x) = \exp(k \sin(x))$$

In my code, I set error tolerance as 10^{-5}. And the return is the minimum N that ensure a maximum error is less thant error tolerance.

- even method

```
$ python appr_derivative.py k e
```

- odd method

```
$ python appr_derivative.py k o
```

And when comparing even method with the odd in different cases, you need to run the following command.

```
$ ./ex1_run.sh
```

Listing 1: ex1_run.sh

```bash
#!/bin/bash
for k in 2 4 6 8 10 12
do
echo "Approximating derivative when k = $k"
python appr_derivative.py $k o
python appr_derivative.py $k e
done
```

Finally we got the following result, and we could conclude that odd method is a little better than even method.

Table 1: the minimum N for each value of k that ensures a maximum error less than 10^{-5}

	2	4	6	8	10	12
odd	20	28	34	42	48	54
even	20	28	36	44	48	56

Problem 2

```
$python ex2.py
```

- a)

  In Figure 1 we plot the maximum pointwise error at $t = \pi$ for an increasing number of grid points for three different methods: second order method, forth order, spectral method. We observed that the higher the order of the method used for approximating the solution at time $t = \pi$, the more accurate it is. The error obtained with N = 2048 using the second-order scheme is the same as that computed using the fourth-order method with N = 128, or the spectral method with only N = 16. In order to avoid time-stepping errors, in my experiment $CFL = \frac{\delta t}{\delta x} \ll 1$.

	2nd	4th	inf
8	1.727338168247775	0.6369004229833811	0.0008431837732436609
16	1.055944124198215	0.20167188861152785	2.132905341234448e-08
32	0.4413218365487239	0.021335556663555355	6.154232679023153e-11
64	0.13150382777251357	0.0014097277016826837	6.309930355996585e-11
128	0.03255250391667852	9.161694042125745e-05	6.338041202980094e-11
256	0.008057882826501661	5.8279711034892046e-06	6.34856611725354e-11
512	0.002012426104225362	3.673636825851645e-07	6.346123626599365e-11
1024	0.0005034368056420213	2.311262070620046e-08	6.34376995378716e-11
2048	0.00012594417006006609	1.506873292811406e-09	6.341638325579879e-11

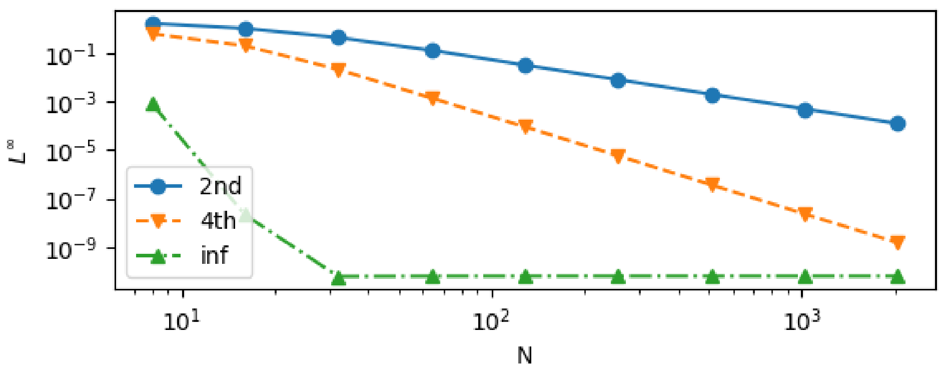

Figure 1: The maximum pointwise error of the numerical solutions, measured at , as a function of N

- b) Figure 2 shows a comparison between the local second-order method and the spectral method for a long time integration. We observed that the global scheme was superior in accuracy to the local scheme, even though the latter scheme employed 20 times as many grid points as the local method did .

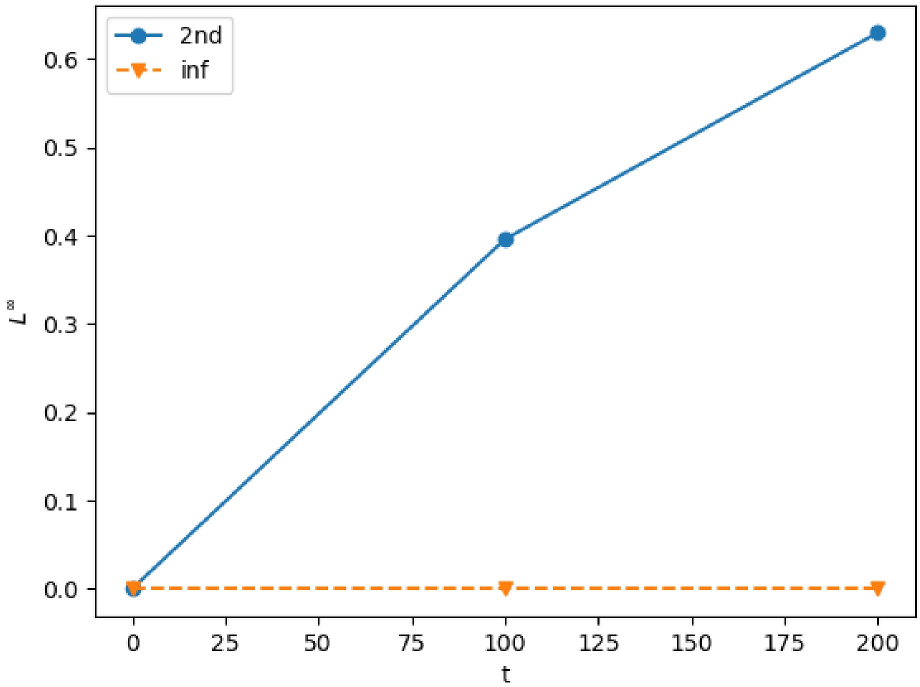

Figure 2: The maximum pointwise error of the numerical solutions, measured at $0, 100, 200$.

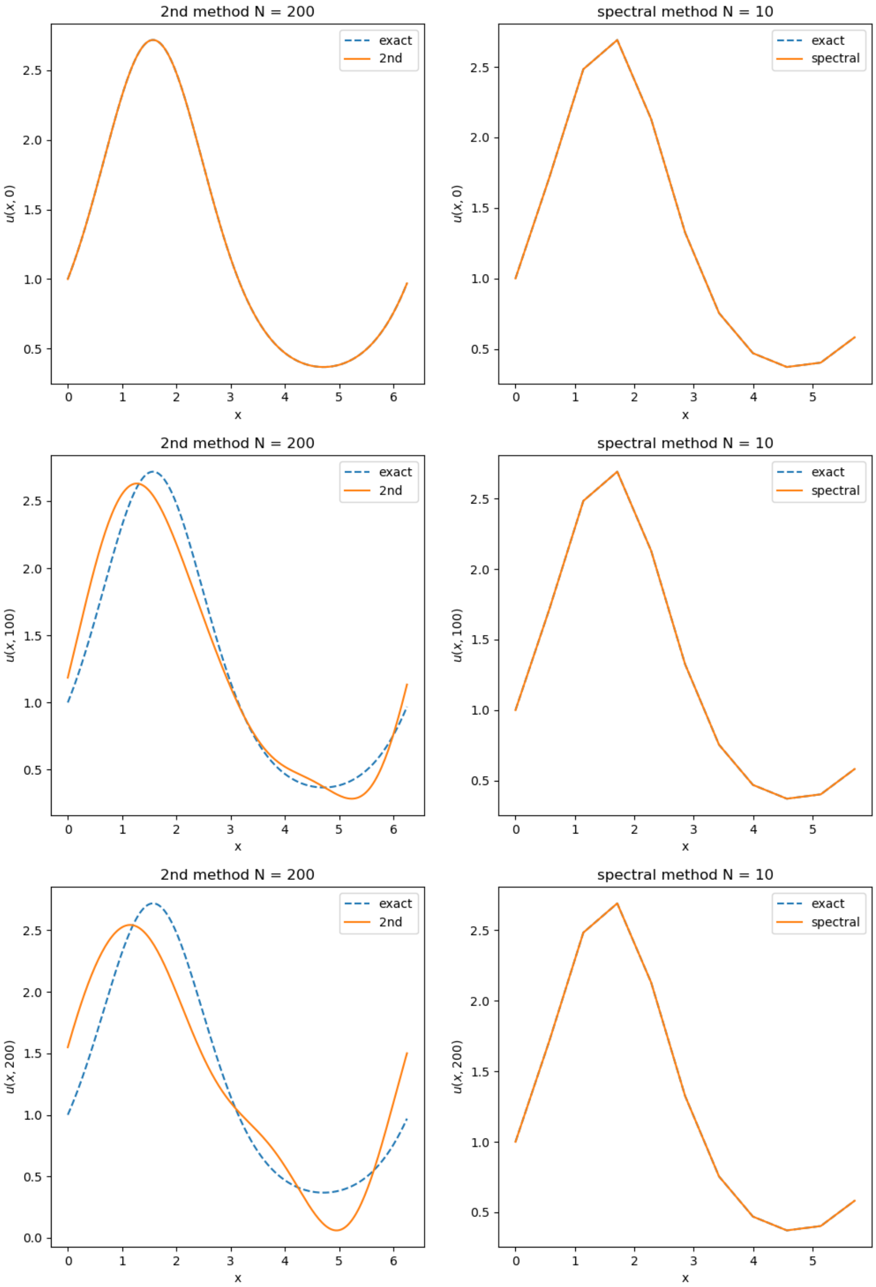

Figure 3: An illustration of the impact of using a global method for problems requiring long time integration. On the left we show the solution computed using a second-order centered-difference scheme. On the right we show the same problem solved using a global method. The full line represents the computed solution, while the dashed line represents the exact solution.