Homework info and requirements

Fast and efficient solvers course

2016

1. General info

Problem sets are distributed in jupyter notebooks. We recommend using Anaconda Python Distribution which allows easy installation of all packages required for the course.

Problem sets include both theoretical problems and coding. Each problem in a problem set should be done in a separate cell. Theoretical problems should be solved inside jupyter notebooks using Markdown[1] mode, which partially supports LaTeX. Coding problems should be solved in Python. Students who are not experienced with Python may find useful tutorials here.

Besides standard problems you will find "bonus problems". They are challenging and non-obligatory. However, they may help you if, for instance, your grade is slightly below higher grade threshold.

Each problem set may have its own number of points. Total score for all problem sets will be then scaled to be 40% of the total grade.

2. Submission details

1. Submission should be a single jupyter notebook named in the following fashion: `surname_name_pset-number.ipynb`, e.g. `oseledets_ivan_1.ipynb`
2. You can submit incomplete solutions. They will be graded as well.
3. Late submissions within 2 days are graded up to 80% of the total score. 50% penalty is applied if you submit even later.
4. If your code produces incorrect results, but is "almost correct", you might be asked to resubmit it with some penalty. Otherwise, you will likely get 0 pts for it.

3. How you should code in Python

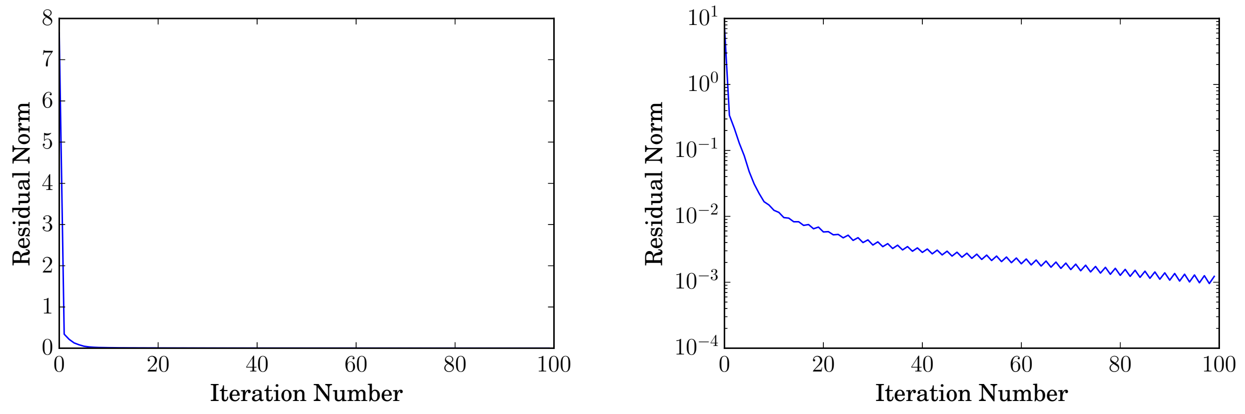
1. Use `numpy` arrays instead of built-in lists.
2. Avoid import *!!! Namespaces is a feature.
3. Try not to use loops in your code as far as it is possible. Vectorized code with `numpy` arrays works much more efficiently!
4. For those of you, who are familiar with MATLAB, we provide a link to manual with correspondence between MATLAB and `numpy` commands[2].
5. We recommend that you follow PEP8 style guide[3].
6. Define variables and functions with meaningful names.
7. Avoid dynamic memory allocation.

[1] http://jupyter-notebook.readthedocs.io/en/latest/examples/Notebook/Working%20With%20Markdown%20Cells.html
[2] https://docs.scipy.org/doc/numpy-dev/user/numpy-for-matlab-users.html
[3] https://www.python.org/dev/peps/pep-0008/

4. How to plot figures and not make TAs angry

Here we list requirements for plots that MUST be satisfied.

1. Every axis should have its own label.
2. Axis labels should have appropriate font size to be readable (12pt is ok).
3. Every figure should have explaining legend.
4. Every figure should have its own title. It should help TAs to understand what a figure is about.
5. Every figure should have appropriate scale. Particularly, you have to choose log-scale or linear scale depending on lines you plot. See Fig. 1 to compare these scales. In this example the log-scale is preferable and plot in linear scale will be penalized.



(a) Linear scale (b) Log-scale

Figure 1: Comparison scales of plots

5. Cite sources you use

Copying someone else's homework or copying solutions from the internet or elsewhere and presenting them as your own is strongly prohibited. If you use other sources (including other students) you must cite them explicitly. However if your work is fully copied (up to renamed variables) or you did not mention a source you will be penalized down to 0 pts for that task.