Report problem1:

1-1: Low rank approximation of matrices

1-2: <u>Matrix product decomposition</u> of tensors (vectors)

* It will be explained in #10 and #11.

- (optional) means, it is not mandatory. So, even if you will not solve the task, in principle, you can get the perfect score. When you will include it, you may get additional points.

- Please include your name and student id in your report.

- Please submit it through **ITC-LMS**
  If you have any troubles, please send us email:
  t-okubo@phys.s.u-tokyo.ac.jp

- **(tentative) Deadline: 2022 Jan. 27th**

Report problem 1-1

Perform the low rank approximation for matrices

(i) Prepare two $M \times N$ matrices A and B. $(M, N > 100$ is better)

It is encouraged to prepare matrices related to your research field.

If it is difficult, prepare two pictures. (It should be different from the examples in the lecture.)

In the following, we compare the low rank approximations of A and B.

So, it is better that A and B have different characteristics (or they are very different pictures).

(In the report, please include the explanation (meaning) of the matrices.)

(ii) Perform SVD and plot the singular values for A and B.

Please normalize the singular values as $\tilde{\sigma}_i = \sigma_i / \sqrt{\sum_j \sigma_j^2}$
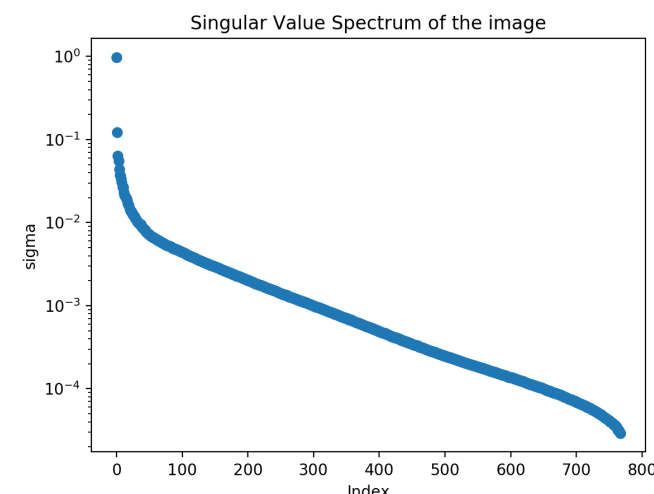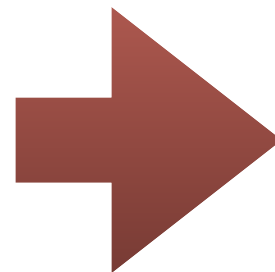
Report problem 1-1 (cont.)

(iii) Perform low rank approximations of the matrices.

- Try more than two ranks for each *A* and *B.*
- Calculate the distances between the original and approximate matrices.
  Please use Frobenius norm $\|A - \tilde{A}\|_F$ as the distance.
  It is better to show a normalized distance: $\|A - \tilde{A}\|_F / \|A\|_F$

(iv) Discuss characteristics of the low rank approximations (for your matrices) based on the singular value spectra. (**This part is the most important!**)

- In the discussion, explain the relation between the distance and singular values.
  (You can find the relation in the lecture slides.)
- If you find a clear difference between the approximations of A and B, please discuss the possible origin of the difference.
  - If the qualities of the approximations look similar, please discuss a candidate matrix which might show a clear difference.
- (optional) Determine the ranks which give normalized distances 10^{-2} and 10^{-3} for your matrices. (You may find a hint in the sample code for "--plot_error" option (in .py) or the last cell in .ipynb.)

Report problem 1-1 (cont.)

> Sample python code for Image SVD: image_svd.py (.ipynb)
>
> (Run with python3. You need PIL, numpy, matplotlib)
>
> I recommend .ipynb (jupyter notebook) and to use Google Colab,
>
> https://colab.research.google.com
>
> where you can run .ipynb from your web browser.

Usage of sample python code for Image SVD:

python image_svd.py -c chi -f filename

[Example of output]

```
Input file: sample.jpg
Array shape: (768, 1024)
Low rank approximation with chi=10
Normalized distance:0.10087303978176487
```

(In addition, the singular value spectrum and the approximated image appear.)

Report problem 1-1 (cont.)

You can see help message: *python image_svd.py -h*

```
usage: image_svd.py [-h] [-c chi] [-f filename]

Low rank approximation of an image

optional arguments:
  -h, --help              show this help message and exit
  -c chi, --chi chi       rank of the approximated matrix. (default: chi = 10)
  -f filename, --file filename
                          filename of the image. (default: sample.jpg)
  --plot_error            plot expected normalized distance. (default: false)
```

Or you can read the text in jupyter notebook.

When you use Google Colab, you need to also upload
image file (e.g. "sample.jpg")
in addition to "image_svd.ipynb".
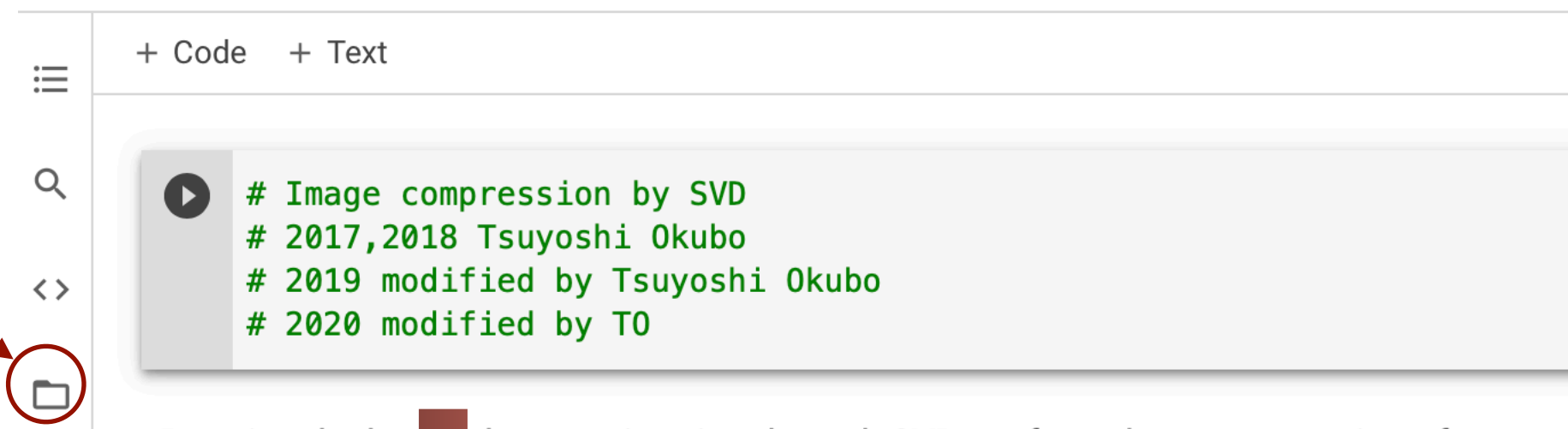
How to use Google Colab

1. Open image_svd.ipynb in Google colab

   ・ Select "**File/upload notebook**" ("ファイル/ノートブックをアップロード") and
   upload image_svd.*ipynb*

   image svd.ipynb ☆
   File   Edit   View   Insert   Runtime   Tools   Help   Last saved at 11:19 PM
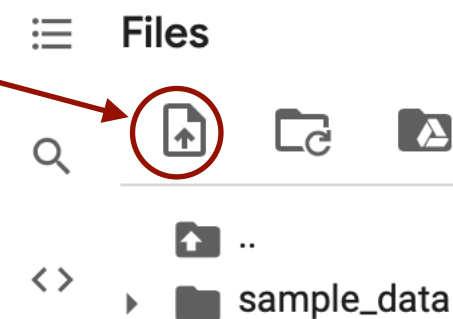
   + Code   + Text

   ```
   # Image compression by SVD
   # 2017,2018 Tsuyoshi Okubo
   # 2019 modified by Tsuyoshi Okubo
   # 2020 modified by TO
   ```

2. Click here

   (Wait a moment
   for the connection)

3. Click here and upload
   your image file (e.g. sample.jpg).

   (Uploaded file will be deleted
   after the session finishes.)

   Files   ✕   + Code

   ..
   ▸ sample_data

4. Select "**Runtime/Restart and Run all**".

Report problem 1-2

Try MPS approximation of vectors (tensors) along the following steps.

1. Prepare two normalized vectors in m^N dimensional vector space. (They can be considered also as N-leg tensors.)

   - If you have a vector related to your research, it is better.
   - If not, I recommend you to try, e.g.,
     - Ground states of the transverse field Ising chain with the open boundary condition. (When you use it, don't forget to mention parameters (value of magnetic field)).
     - A picture, which is reshaped properly, is another candidate. For example, we easily reshape 128 x 128 pixel image to 2^{14} dimensional vector $((m, N) = (2, 14))$.
     - A random vector. (**Please don't use two random vectors.**)

2. Calculate the Schmidt coefficients for a bipartition $m^{N/2} \otimes m^{N/2}$, and entropies of them (entanglement entropy).

   - Note: Schmidt coefficient is related to the singular values.
   - Please include figures of Schmidt coefficients spectra, and values of the entropies.
   - For the transverse field Ising chain, please consider the bipartition at the center.
   - For pictures, please write down how you divided the vector.

Report problem 1-2 (cont.)

4. Make the exact MPS from it. Then, by using low rank approximation based on SVD, make approximate MPS with (maximum) bond dimension chi_max.
   - Note: After this approximation, the norm of a vector becomes smaller.

5. Calculate distances between exact and approximate MPSs

$$\left\| \vec{v}_{ex} - \vec{v}_{ap} \right\|$$

   by varying chi_max and discuss their behaviors.
   - Please mention the above Schmidt coefficient spectra in the discussion.
   - Consider differences between two prepared vectors.
   - Consider a relationship to the low rank approximation in problem 1-1.

6. (optional) By varying N, discuss the change of the distances. Please plot the entanglement entropy as a function of N, and consider possible relationship to the behavior of the distances.

7. (optional) Change the order of indices (exchange the position of tensor legs), and discuss the quality of the approximations.
   - In the case of pictures, the transformation to a tensor in the sample code might not be the best.

Report problem 1-2: sample codes

When you consider spin models, these steps can be done by sample python code Report_Spin.py (or .ipynb)

Usage: python Report_Spin.py -N N -m m -chi chi_max -Jz Jz -Jxy Jx -hx hx

```
okubo$ python Report_Spin.py
Model parameters: Jz, Jxy, hx = -1.0, 0.0, 0.5
Parameters: N, m, chi_max = 16, 2, 2
Ground state energy per bond = -0.33360646500808594
Energy of Exact MPS = -0.33360646500808566
Truncation: chi_max = 2
Energy of MPS with truncation = -0.3327045663989325
Energy difference: E_truncated - E_exact =0.0009018986091531844
Distance between exact and truncated MPS = 0.10640004821106396
Entanglement entropy = 0.4234093173532134
```

$$\mathcal{H} = J_z \sum_{i=1}^{N-1} S_i^z S_{i+1}^z + J_{xy} \sum_{i=1}^{N-1} (S_i^x S_{i+1}^x + S_i^y S_{i+1}^y) - h_x \sum_{i=1}^{N} S_i^x$$

When we set

$J_z = -1, J_{xy} = 0, h_x \neq 0, m = 2$ ➡ Transverse filed Ising chain

$J_z = 1, J_{xy} = 1, h_x = 0$ ➡ Heisenberg chain with $S = (m-1)/2$

Report problem 1-2: sample codes (cont.)

In the case of a picture image, you can use Report_Image.py (.ipynb).

Usage: python Report_Image.py -N N -m m -chi chi_max -f $filename$

In this code, the original (X, Y) pixel image is transformed into the gray scale and the center $(m^{N/2}, m^{N/2})$ region is used as the m^N dimensional vector.

output:

```
okubo$ python Report_Image.py
Parameters: N, m, chi_max = 16, 2, 20
Input file: sample.jpg
Original size: (1024, 768)
Trimmed size: (256, 256)
Truncation: chi_max = 20
Distance between exact and truncated MPS = 0.18728957286061443
Entanglement entropy = 0.9494402492802109
```
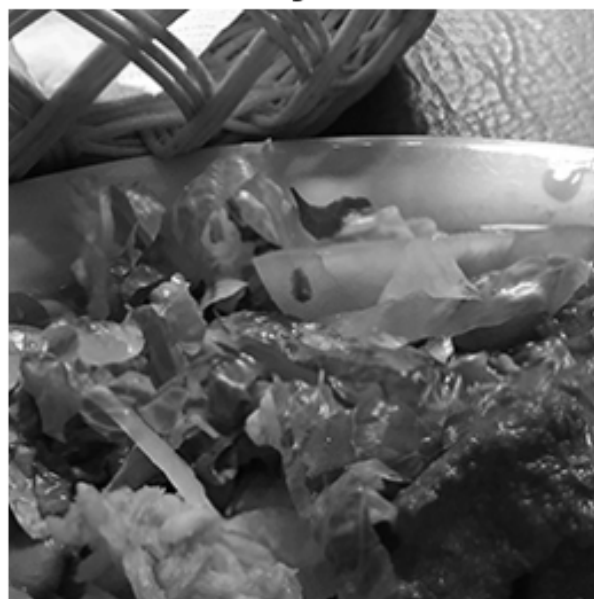

original   approximated

Report problem 1-2: sample codes (cont.)

You can also use Report_Random.py (or .ipynb) for a random vector.

Usage: python Report_Random.py -N N -m m -chi chi_max -s $seed$

output:
```
okubo$ python Report_Random.py
Parameters: N, m, chi_max = 10, 2, 20
Random seed: = None
Truncation: chi_max = 20
Distance between exact and truncated MPS = 0.20232676589868706
Entanglement entropy = 2.941107062678152
```

In sample codes, you can see help message by, e.g.,

python Report_Random -h

(Also you can read the text in jupyter notebook.)

When you use Google Colab, you need to also upload
"ED.py", "MPS.py" and "Report_Random.py"
for the case of "Report_Spin.ipynb", and
"Report_Random.py" and your image file,
for the case of "Report_Image.ipynb".

Report problem 1-2: sample codes (cont.)

So far, I have prepared sample codes for the spin model, picture images, and random vectors. However, if you have another example which you will try to apply MPS decomposition, please freely contact me via email:

t-okubo@phys.s.u-tokyo.ac.jp

I will consider to write a sample (python) code corresponding to your example.

If you have any troubles or questions, also please contact me freely.