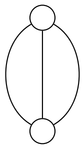
Christian B. Mendl, Richard M. Milbradt due: 19 May 2022 (before tutorial)

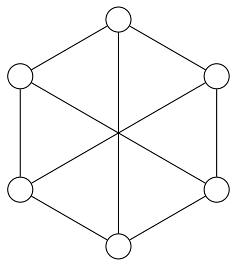
Tutorial 3 (Counting graph colorings)
Consider the following problem: given a 3-regular graph (i.e., a graph where each vertex has three connected edges), how many edge colorings using three colors exist, such that the edges at each vertex have distinct colors?

(a) Explicitly enumerate the allowed edge colorings for the following graph:



(b) Interpreting a 3-regular graph as tensor network, with each vertex a tensor of degree 3, how can we define these tensors such that contracting the tensor network yields the number of edge colorings?

(c) Compute how many ways one can color the following graph:



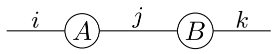
As a remark, it turns out that if the vertices are described by the *Levi-Civita symbol* ϵ defined as

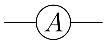
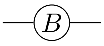
$$\epsilon_{ijk} = \begin{cases} +1 & \text{if } (i,j,k) \text{ is an even permutation of } (1,2,3) \\ -1 & \text{if } (i,j,k) \text{ is an odd permutation of } (1,2,3) \\ 0 & \text{otherwise} \end{cases}$$

the contraction will count the colorings for planar graphs, but yield 0 for non-planar ones.[1] You can test this statement for the graph above. (As voluntary homework puzzle, try to prove this statement in general.)

Exercise 3.1 (Tensor diagrams)

(a) Recall that the matrix product AB with entries $(AB)_{ik} = \sum_j a_{ij} b_{jk}$ translates to the following diagram:

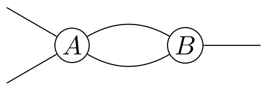


Connect the legs of  and  accordingly to represent AB^T, $B^T A^T$ and $\mathrm{tr}[AB]$ graphically.

(b) Given a matrix A, a tensor B of degree 3, and a vector C, express the following tensor contraction in graphical form:

$$m_{ik} = \sum_{j,\ell} a_{ij}\, b_{kj\ell}\, c_\ell.$$

(c) Let A and B be tensors of degree d_A and d_B, respectively, such that each individual dimension is equal to some $n \in \mathbb{N}$. (In other words, $A \in \mathbb{C}^{n\times\cdots\times n}$ where n appears d_A times, and likewise for B.) Now A and B are contracted along c of these dimensions, as illustrated for $d_A = 4$, $d_B = 3$ and $c = 2$ in the following diagram:



What is the asymptotic computational cost of this contraction (in the form $\mathcal{O}(n^\ell)$ with to-be determined exponent ℓ) based on a literal implementation of the summation formulation?

Hint: Determine the required number of nested for-loops from 1 to n to compute the entries of the resulting tensor.

[1]See Roger Penrose: *Applications of negative dimensional tensors.* Combinatorial Mathematics and its Applications (1971)

Exercise 3.2 (Matrix functions)

(a) Compute $\mathrm{e}^{-\beta X}$, with $\beta \in \mathbb{R}$ and $X = \left(\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right)$ the Pauli-X matrix.

Hint: Find the spectral decomposition of X first. To simplify the final expression, the relations $\frac{1}{2}(\mathrm{e}^{\beta} + e^{-\beta}) = \cosh(\beta)$ and $\frac{1}{2}(\mathrm{e}^{\beta} - e^{-\beta}) = \sinh(\beta)$ might be helpful.

(b) Now insert $\beta = 0.4$ into your solution of (a); it should agree with the result of the following Python code:

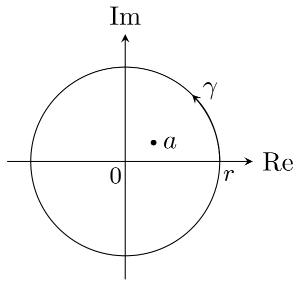
```
import numpy as np
from scipy.linalg import expm

# Pauli-X matrix
X = np.array([[0., 1.], [1., 0.]])

# evaluate matrix exponential numerically
beta = 0.4
expm(-beta * X)
```

Hint: You can compute $\cosh(\beta)$ and $\sinh(\beta)$ numerically via `np.cosh(beta)` and `np.sinh(beta)`.

(c) Cauchy's integral formula for a holomorphic (complex differentiable) function $f : \mathbb{C} \to \mathbb{C}$ states that, for any $a \in \mathbb{C}$,



$$f(a) = \frac{1}{2\pi i} \oint_\gamma \frac{f(z)}{z-a} \mathrm{d}z, \tag{1}$$

where the integration contour γ can be chosen as circle in the complex plane which encloses a. When parametrizing $z = r\mathrm{e}^{2\pi i t}$ with r the radius of this circle and $t \in [0, 1]$, and using that $\frac{\mathrm{d}z}{\mathrm{d}t} = 2\pi i r \mathrm{e}^{2\pi i t}$ for a change of variables, the contour integral becomes

$$\frac{1}{2\pi i} \oint_\gamma \frac{f(z)}{z-a} \mathrm{d}z = \int_0^1 \frac{f(r\mathrm{e}^{2\pi i t})}{r\mathrm{e}^{2\pi i t} - a} r\mathrm{e}^{2\pi i t} \mathrm{d}t.$$

The following Python code is a demonstration for $f(z) = \mathrm{e}^{-\beta z}$ (recall that the imaginary unit is denoted `j` in Python):

```
import numpy as np
from numpy import exp
import scipy.integrate as integrate

beta = 0.4
def f(z):
    return exp(-beta*z)

a = 0.3 + 0.2j

r = 2.0  # circle radius
integrand = lambda t: f(r*exp(2j*np.pi*t)) / (r*exp(2j*np.pi*t) - a) * r*exp(2j*np.pi*t)
# perform integration of real and imaginary parts separately
(   integrate.quad(lambda t: np.real(integrand(t)), 0, 1)[0] +
 1j*integrate.quad(lambda t: np.imag(integrand(t)), 0, 1)[0])
```

This gives $0.884 - 0.0708i$ (for the leading digits) and agrees with $f(a)$, as it should be according to (1).

It turns out that we can use Cauchy's integral formula[2] to evaluate matrix functions as well. Namely, for normal $A \in \mathbb{C}^{n \times n}$,

$$f(A) = \frac{1}{2\pi i} \oint_\gamma f(z)(zI - A)^{-1} \mathrm{d}z, \tag{2}$$

where I denotes the identity matrix. The contour γ now has to enclose all eigenvalues of A. As advantage for numerical calculations, this formulation bypasses an explicit spectral decomposition of A, but instead requires to compute the so-called *resolvent* $(zI - A)^{-1}$, that is, the inverse matrix of $zI - A$.

Modify the above Python code such that it computes the matrix-valued Cauchy integral in Eq. (2) for $A = X$, and evaluate the integral for $f(z) = \mathrm{e}^{-\beta z}$ with $\beta = 0.4$. The result should agree with (a) and (b).

Hint: Replace `quad` by `quad_vec`, and use the relation $(zI - X)^{-1} = \frac{1}{z^2-1}(zI + X)$.

[2]Complex analysis only appears in this exercise and is not relevant for the final exam.