Exercise sheet 4
Theoretical Nanooptics WS 2023/2024

Karlsruhe Institute of Technology
Ivan Fernandez-Corbaton
Ivanina Ilieva

February 9th 2024

1 True or false?

State whether the following statements are true or false and write a short reasoning for your answer.

a) (1 Point) Duality transformation commutes with rotations around any axis.

b) Consider the state $|\Phi\rangle = |\mathbf{k}\,+\rangle + |\mathbf{k}\,-\rangle$:

   i) (1/2 Point) $|\Phi\rangle$ is an eigenstate of the helicity operator.

   ii) (1/2 Point) $|\Phi\rangle$ is an eigenstate of the parity operator.

   iii) (1 Point) $|\Phi\rangle$ can be an eigenstate of infinitely many mirror reflection operators.

c) (1/2 Point) A rotation commutes with all mirror reflections.

d) (1/2 Point) A rotation never commutes with a mirror reflection.

e) (1/2 Point) Parity followed by a mirror reflection anti-commutes with the helicity operator.

f) Consider a dual-symmetric object with a discrete rotational symmetry $R_z\left(\frac{2\pi}{4}\right)$

   (a) (1/2 Point) The illumination $|k\hat{\mathbf{z}}\,+\rangle + |2k\hat{\mathbf{z}}\,-\rangle$ produces some backscattering.

   (b) (1 Point) Such an object makes a good light sail, where a laser pushes the object away by means of the optical force.

2 treams

In this task, you will go through a short introductory example of solving a physical problem using the code `treams`, which is a T-Matrix scattering code for nanophotonic computations developed by the Institute of Theoretical Solid State Physics at KIT [1]. `treams` is a python package, which you can install on your machine for this exercise. You can find the installation guide in the documentation https://tfp-photonics.github.io/treams/ along with other useful information you might need for this task. You do not really need any programming experience for this exercise, just the hints given and the code documentation. When in doubt, remember that there is a command implemented for everything that we are looking to calculate here.

Please submit a .py or .ipynb file together with your pen and paper submission. Make sure your code is clean and easily readable, well commented such that each step is logically traceable, and print all of the sought after values together with some text that unambiguously specifies what they are.

In case you are not familiar with Python, you can still solve this exercise. You do not really need any Python syntax except for importing packages, assigning variables and printing lines and variables, which is sketched in a simple example below. Everything else is `treams`-specific syntax you can find in the documentation.

In case you need it, there is an official Python tutorial here: https://docs.python.org/3/tutorial/index.html. Python can also be used within the framework of Conda/Miniconda, which is a command line tool that supports and manages environments and has a user friendly interface. You can find the user guide here: https://conda.io/projects/conda/en/latest/user-guide/index.html.

```
# python example
import treams
import numpy as np

# assigning values to some variables
a = 4-3j
b = -2+1j

# calculating some things and printing
abs_a = np.abs(a)
print("Absolute value of a is ", abs_a)
print("Absolute value of b is ", np.abs(b))
print("a(b*)=", a*np.conjugate(b))
```

This code returns the following

```
Absolute value of a is 5.0
Absolute value of b is 2.23606797749979
a(b*)= (-11+2j)
```

You will be calculating some T-matrices in this exercise, so let us first shortly discuss what the T-matrix actually is. In the lecture, you have learned about the scattering matrix, the S-matrix, which is closely connected to the T-matrix through

$$S = \mathbb{1} + 2T \ . \tag{1}$$

The S-matrix in that sense relates the incoming fields to the outgoing ones.

When considering how light interacts with matter, one can always just look at the incident electric fields and upon calculation of the scattered electric fields, all of the relevant information about the waves outside of the scatterer is known. The magnetic fields can be calculated from the electric ones in accordance to Maxwell's equations. If we expand the incoming and scattered fields in the spherical wave basis, we get four coefficients, a_{lm}, b_{lm}, p_{lm} and q_{lm}, in agreement with [2, Eqs. (5-6)], Eqs. (5), (6)

$$\begin{aligned} \mathbf{E}^{\mathrm{inc}}(\mathbf{r}) &= \sum_{l=1}^{\infty} \sum_{m=-l}^{l} \left(a_{lm}\mathbf{M}_{lm}(\mathbf{r}) + b_{lm}\mathbf{N}_{lm}(\mathbf{r}) \right) \\ \mathbf{E}^{\mathrm{sca}}(\mathbf{r}) &= \sum_{l=1}^{\infty} \sum_{m=-l}^{l} \left(p_{lm}\mathbf{M}_{lm}(\mathbf{r}) + q_{lm}\mathbf{N}_{lm}(\mathbf{r}) \right) \ , \end{aligned} \tag{2}$$

where $\mathbf{M}_{lm}(\mathbf{r})$ and $\mathbf{N}_{lm}(\mathbf{r})$ are the vector spherical harmonics, and with $\mathbf{r}$ pointing outside of the scatterer. Since Maxwell's equations are linear, one can relate the incident field coefficients, a_{lm} and b_{lm}, to the scattered field coefficients, p_{lm} and q_{lm} via a linear transformation, which can be formulated as a matrix. Calling this matrix T, we can write down the relation as

$$\begin{pmatrix} \mathbf{p} \\ \mathbf{q} \end{pmatrix} = T \begin{pmatrix} \mathbf{a} \\ \mathbf{b} \end{pmatrix} \tag{3}$$

In a linear, time-invariant, homogeneous, isotropic, local and reciprocal medium the chiral constitutive relations are given by

$$\begin{pmatrix} \frac{1}{\epsilon_0}\mathbf{D} \\ c\mathbf{B} \end{pmatrix} = \begin{pmatrix} \epsilon & \mathrm{i}\kappa \\ -\mathrm{i}\kappa & \mu \end{pmatrix} \begin{pmatrix} \mathbf{E} \\ Z_0\mathbf{H} \end{pmatrix} \ , \tag{4}$$

where ϵ, μ and κ are the relative permittivity, relative permeability, and chirality parameter, respectively. In case you are not familiar with them, there is some useful information on cross sections on Wikipedia: https://en.wikipedia.org/wiki/Cross_section_(physics).

a) (2 Points) Define a chiral sphere in a chiral embedding and calculate the rotational and polarization average scattering and extinction cross-section for this setting. You will need to calculate the T-matrix as an intermediate step.
Use the following system parameters: Sphere - relative permittivity $\epsilon = 4 + 0.1\mathrm{i}$, relative permeability $\mu = 1 + 0.1\mathrm{i}$, chirality parameter $\kappa = 0.5 + 0.05\mathrm{i}$; Embedding medium - relative permittivity $\epsilon = 1.2$, relative permeability $\mu = 1$, chirality parameter $\kappa = 0.1$. Use a maximal multipolar order of 3 for your T-matrix, a wavelength in vacuum $\lambda_0 = 1$ and a sphere radius $r = 0.3$. Note that in the code the units are implemented relatively and hence, here they are arbitrary, meaning that as long as all parameters given are of the same order of magnitude (say, everything is in nm), one does not need to specify it.
Hint: For setting up the materials for the sphere and the embedding, use the class `treams.Material`, for the creation of the sphere and calculation of its T-matrix use the classmethod `treams.TMatrix.sphere` and finally, for the rotational and polarization average scattering and extinction cross-section use the property `treams.TMatrix.xs_sca_avg` and `treams.TMatrix.xs_ext_avg`, respectively.

b) (2 Points) Calculate the scattering and extinction cross section of the same chiral sphere as above now depending on the illumination direction and polarization. Implement two incoming plane waves, each propagating along the z-axis with vacuum wavelength $\lambda_0 = 1$ and helicity $+1$ and -1, respectively, and calculate the cross sections for these waves as illumination.
Hint: To implement the plane waves, use the function `treams.plane_wave` (this gives you the wave coefficients), where you can assign the helicity to the argument *pol* (0 or 1, see Documentation for more) after checking that the default polarization mode is *helicity* (in `treams.config.POLTYPE`). Use the property `treams.TMatrix.xs` to calculate the cross section for the specific illumination wave.

c) (2 Points) Create a dimer out of two of the chiral spheres implemented above positioned at $\mathbf{r}_1 = (0, 0, -0.4)$ and $\mathbf{r}_2 = (0, 0, 0.4)$. In a setting such as this, upon illumination, there will be multiscattering between the two spheres. Hence, one can incorporate this coupling in the off-diagonal elements of the T-matrix and then work with this T-matrix, which now describes the dimer of the two spheres as a whole. Construct the so-called local T-matrix of the composite system by letting the two spheres interact and calculate the scattering and extinction cross section of the dimer for illumination with a plane wave propagating along the z-axis with vacuum wavelength $\lambda_0 = 1$ and helicity $+1$.
Hint: Use the classmethod `treams.TMatrix.cluster` to create the block-diagonal T-Matrix that includes all the individual T-matrices of the cluster elements (here, the two spheres). Going from that to the local T-matrix that includes the coupling happens via the function `treams.TMatrix.interaction.solve()`. This function does not need any arguments, just use it as an attribute of the block-diagonal T-matrix you get from the `cluster` method.

d) (2 Points) The local T-matrix of a composite system, as the name suggests, is an expansion of the contributions into the local bases, which is a set of bases, each centered around one of the objects in the system. In this example, we have a set of two spherical wave bases, each centered around one of the spheres. Logically, it would be a lot simpler to describe the system using just one common basis. Expand the local T-matrix of the dimer from above into a global spherical wave basis centered around the origin of the coordinate system up to a multipolar order of 10 and calculate the scattering and extinction cross section for an illumination with a plane wave propagating along the z-axis with vacuum wavelength $\lambda_0 = 1$ and helicity $+1$. Compare the values for the cross sections calculated with the local and global T-matrix.
Hint: Using the classmethod `treams.SphericalWaveBasis.default`, define the global spherical wave basis and expand the local T-matrix into it via the function `treams.TMatrix.expand`.

e) (2 Points) Rotate the dimer from above by an angle of $\pi/2$ about the y-axis and calculate again the T-matrix and scattering and extinction cross section for illumination with a plane wave propagating along the z-axis with vacuum wavelength $\lambda_0 = 1$ and helicity $+1$. Have the cross section values changed? Why?
Hint: Define a rotation operator with the appropriate Euler angles (`treams.Rotate`, which returns a class *Operator* object) and calculate the T-matrix after the rotation using the active transformation law for an operation R on a matrix T, RTR^{-1}. Build the inverse by utilizing the built-in method `treams.operators.inv`. For matrix multiplication use the Python @ operator. After applying the transformation on the original T-matrix, you can define the new T-matrix object by using the class `treams.TMatrix`.

f) (2 Points) Build a square lattice out of chiral spheres such as the ones above, with a lattice constant $a = 0.8$. Compute the S-matrix (use the basis of the plane wave from above for this) and then calculate the transmittance and reflectance for illumination with a plane wave propagating along the z-axis with vacuum wavelength $\lambda_0 = 1$ and helicity $+1$ and -1. Do not forget to let the spheres in the lattice interact and build the S-matrix from the corresponding T-matrix that includes the coupling Do you notice something peculiar about the coefficients? Can you explain it?
Hint: First define the lattice by using an appropriate method of the implemented class `treams.Lattice`, then calculate the interaction T-matrix of the lattice by utilizing the method `treams.TMatrix.latticeinteraction.solve`, which takes the arguments *(lat, kvec)*, where *lat* is a `treams.Lattice` object and *kvec* is an array with the **k**-vector coefficients that the lattice is to be illuminated by. Then, the classmethod `treams.SMatrices.from_array` will give you the S-matrix as an *SMatrices* object, which you can use to calculate the transmittance and reflectance using `treams.SMatrices.tr`. You can extract the basis of a given array by utilizing the property `.basis`

g) (1 Point) Perform the same calculation but for a grid that is slightly off from square, i.e. for a rectangular grid with lattice constants 0.79 and 0.81. In this setting one could argue that the rotational four-fold symmetry of the lattice is "approximate". Would you say that is the case here? Look at the new values you get. How does the result change? Can you say something about the amount by which it changes?

h) (1 Point) Run again the same calculation for a rectangular grid with distinctly different lattice constants, 0.8 and 1.6, and compare your results to those from the previous computations.

References

[1] Beutel, D., Fernandez-Corbaton, I., and Rockstuhl, C. (2023). treams–A T-matrix scattering code for nanophotonic computations. *arXiv preprint arXiv:2309.03182.* https://arxiv.org/abs/2309.03182

[2] Mishchenko, M., Travis, L., and Mackowski, D. 1996. T-matrix computations of light scattering by nonspherical particles: A review, Journal of Quantitative Spectroscopy and Radiative Transfer, Vol. 55, https://www.sciencedirect.com/science/article/pii/0022407396000027