Progress report:

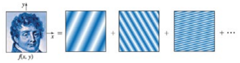
Optimization for 1D Scalar Grating Splitting 2 Beams

Yichen Zhu

Review

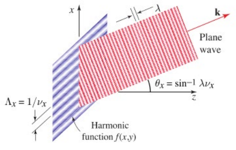
1 Scalar Fourier Optics

- complex scalar **transmittance function**

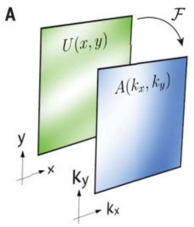


$$U(x,y) = t(x,y)E_0(x,y)$$

- **plane wave expansion**



$$\begin{cases} U(x,y) = \iint_{-\infty}^{+\infty} A(k_x,k_y)e^{-i(k_x x+k_y y)}d\frac{k_x}{2\pi}d\frac{k_y}{2\pi} \\ A(k_x,k_y) = \iint_{-\infty}^{+\infty} U(x,y)e^{i(k_x x+k_y y)}dxdy \end{cases}$$



→ **xy coordinate space** v.s. **k wave vector space**

- requirements: **paraxial incident**

2 Vector Fourier Optics

- Matrix transmittance function (**local Jones Matrix**) $\tilde{J}(x,y)$
- vector valued incident light (**local Jones Vector**) $|E_0(x,y)\rangle$

$$|U(x,y)\rangle = \tilde{J}(x,y)|E_0(x,y)\rangle$$

- **plane wave expansion**

$$\begin{cases} |U(x,y)\rangle = \tilde{J}(x,y)|E_0(x,y)\rangle = \iint_{-\infty}^{+\infty} |A(k_x,k_y)\rangle\, e^{-i(k_x x+k_y y)}d\frac{k_x}{2\pi}d\frac{k_y}{2\pi} \\ |A(k_x,k_y)\rangle = \iint_{-\infty}^{+\infty} \tilde{J}(x,y)|E_0(x,y)\rangle e^{i(k_x x+k_y y)}dxdy \end{cases}$$

- requirements: **paraxial incident**

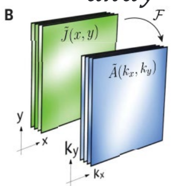
3 Matrix Fourier Optics

- special case 1:

normal, uniform incident

$$\tilde{J}(x,y)|E_0(x,y)\rangle \rightarrow \tilde{J}(x,y)|E_0\rangle$$

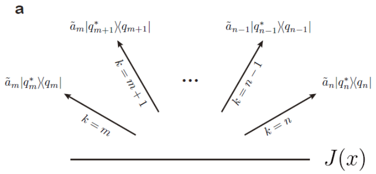
$$\begin{cases} \tilde{J}(x,y) = \iint_{-\infty}^{+\infty} \tilde{A}(k_x,k_y)e^{-i(k_x x+k_y y)}d\frac{k_x}{2\pi}d\frac{k_y}{2\pi} \\ \tilde{A}(k_x,k_y) = \iint_{-\infty}^{+\infty} \tilde{J}(x,y)e^{i(k_x x+k_y y)}dxdy \end{cases}$$



- special case 2:

normal, uniform incident periodic grating

$$\begin{cases} \tilde{J}(x,y) = \sum_{\vec{k}} \tilde{J}_{\vec{k}}\, e^{-i(k_x x+k_y y)} \\ \tilde{J}_{\vec{k}} = \iint_{-\infty}^{+\infty} \tilde{J}(x,y)e^{i(k_x x+k_y y)}dxdy \end{cases}$$



4 Beam Splitting → Parallel Polarization Analyzer

- **multiplexing** in **k space: polarization analyzer**

$$\tilde{J}(x,y) = \sum_{\vec{k}\in\{l\}} \tilde{J}_{\vec{k}}\, e^{-i(k_x x+k_y y)}, \qquad \text{where } \tilde{J}_{\vec{k}} = a_k|p_k\rangle\langle q_k|$$

- **realization** in **xy space: pp phase metasurface**

$$\tilde{J}(x,y) = R(\theta(x,y))\begin{bmatrix} e^{i\varphi_x(x,y)} & 0 \\ 0 & e^{i\varphi_y(x,y)} \end{bmatrix}R(-\theta(x,y))$$

- **conflict between two space:**

can only **ideally** separate **conjugate polarized light**

→allow light **to leak into other orders**

Analytical Problem → **Optimization Problem**

Optimization for Parallel Polarization Analyzer

Outline:

Optimization for Parallel Polarization Analyzer

- 1) 1D Scalar Grating
  - Diffraction Grating
    - Traditional Diffraction Grating
    - No-gain Diffraction Grating
  - Phase-only Grating
    - Conflict Between x-space and k-space
    - Phase-only Grating Optimization
      - The Constrained Optimization Problem
      - The Least Squares Optimization Problem
      - The Minimum Variance Optimization Problem
    - Upper Bound on the Efficiency
      - The Generalized Constrained Optimization Problem
      - Numerical Results
  - Special Case
    - Non-trivial Analytical Solution: Two Beam Spliting
    - New Restrictions on Gratings: Binary Gratings
    - The Effect of Symmetry: Even Orders and Odd Orders Gratings
    - One Dimensional Continuous Gratings
- 2) 2D Scalar Grating
  - Two Dimensional Fourier Series on General Lattices
  - Two Dimensional Symmetry
  - Two Dimensional Beam Splitting
    - The General Theory
    - Symmetry and Symmetry Breaking
    - Examples Using Square Gratings
      - Four Beam Splitting Using Square Gratings
      - Five Beam Splitting Using Square Gratings
      - Nine Beam Splitting With Square Gratings
      - Examples Using Hexagonal Gratings
      - Six Beam Splitting
      - Seven Beam Splitting with Hexagonal Gratings
  - Two Dimensional Continuous Gratings
- 3) 2D Matrix Grating
  - Parallel Polarization Analysis by Matrix Gratings
    - Tetrahedron Grating
    - Cctahedron grating
  - Parallel Polarization Analysis by Linearly Birefringent Matrix Gratings
    - Linear Birefringence Implies |pk> = |qk∗>
    - Relation to Past Work on “Metasurface Polarization Optics”
    - A Unitary, Linearly Birefringent Matrix Grating for Polarization Analysis Can Only Have Two Orders Acting as Analyzers
  - Optimization-based Design Of Matrix Gratings For Parallel Polarization Analysis
    - Development Of An Optimization Scheme
      - Parameterization Of The Grating Unit Cell
      - Optimization
    - Results for 11-element Gratings
      - Progression of the Optimization
      - Optimized Results
    - Conversion to Metasurface Design

Reference

Start from 1D scalar grating splitting 2 Beams, which has analytical optimization solution:

Optimization Problem Description:

Find the real 2π periodic function ϕ(x) such that we maximize the energy: $E = \frac{|a_1|^2 + |a_{-1}|^2}{\sum_{k=-\infty}^{\infty} |a_k|^2}$

subject to the constraint that: $|a_1|^2 = |a_{-1}|^2$

where a_k are the Fourier coefficients of $e^{i\phi(x)}$.

Theoretical Solution:

$$a_1 = a_{-1} = \frac{2}{\pi} \qquad \eta = |a_1|^2 + |a_{-1}|^2 = \frac{8}{\pi^2} \approx .8106$$

Reference:

Gori, F. (1997). Diffractive optics: An introduction. In S. Martellucci, & A. N. Chester (Eds.), Diffractive optics and optical microsystems. New York: Plenum.

Optimization for Parallel Polarization Analyzer

My code and idea:

```
import numpy as np
import torch
import matplotlib.pyplot as plt


x_values = np.linspace(-np.pi, np.pi, 10000)
p_values = x_values
x = torch.tensor(x_values, dtype=torch.float32)
p = torch.tensor(p_values, dtype=torch.complex64)
p.requires_grad = True

def compute_fourier_coefficients(p, x):
    f_x = torch.exp(1j * p)
    N = len(x)
    coefficients = torch.fft.fft(f_x) / N
    return coefficients

def energy_efficiency(coefficients):
    total_energy = torch.sum(torch.abs(coefficients)**2)
    abs_square_coefficient_pos1 = torch.abs(coefficients[1])**2
    abs_square_coefficient_neg1 = torch.abs(coefficients[-1])**2
    efficiency_ratio = (abs_square_coefficient_pos1 + abs_square_coefficient_neg1) / total_energy
    return efficiency_ratio

def eval(coefficients, alpha=0.5):
    abs_square_coefficient_pos1 = torch.abs(coefficients[1])**2
    abs_square_coefficient_neg1 = torch.abs(coefficients[-1])**2
    balance_term = abs(abs_square_coefficient_pos1 - abs_square_coefficient_neg1)
    return (1 - alpha) * (1 - energy_efficiency(coefficients)) + alpha * balance_term
```

 # take period=2pi for convenience, it doesn't affect Fourier coefficients

 # 1D Torch.tensor "p" describes phase function $\phi(x)$

 # FFT calculates coefficients

 # energy efficiency:

$$E = \frac{|a_1|^2 + |a_{-1}|^2}{\sum_{k=-\infty}^{\infty} |a_k|^2}$$

 # eval function:

(1-α)(1-E)+α(|a_1|^2-|a_-1|^2)

Optimization for Parallel Polarization Analyzer

My code and idea:

```
learning_rate = 2.0
eval_values = []

for epoch in range(30000):
    coefficients = compute_fourier_coefficients(p, x)
    e = eval(coefficients)
    e.backward()
    eval_values.append(e.item())  # Save the eval value
    with torch.no_grad():
        p -= learning_rate * p.grad
        p.grad.zero_()
    if epoch % 100 == 0:  # Print every 100 epochs
        print('Epoch:', epoch, 'Eval:', e.item())

print("Result:", p)
```

Pytorch grad descent

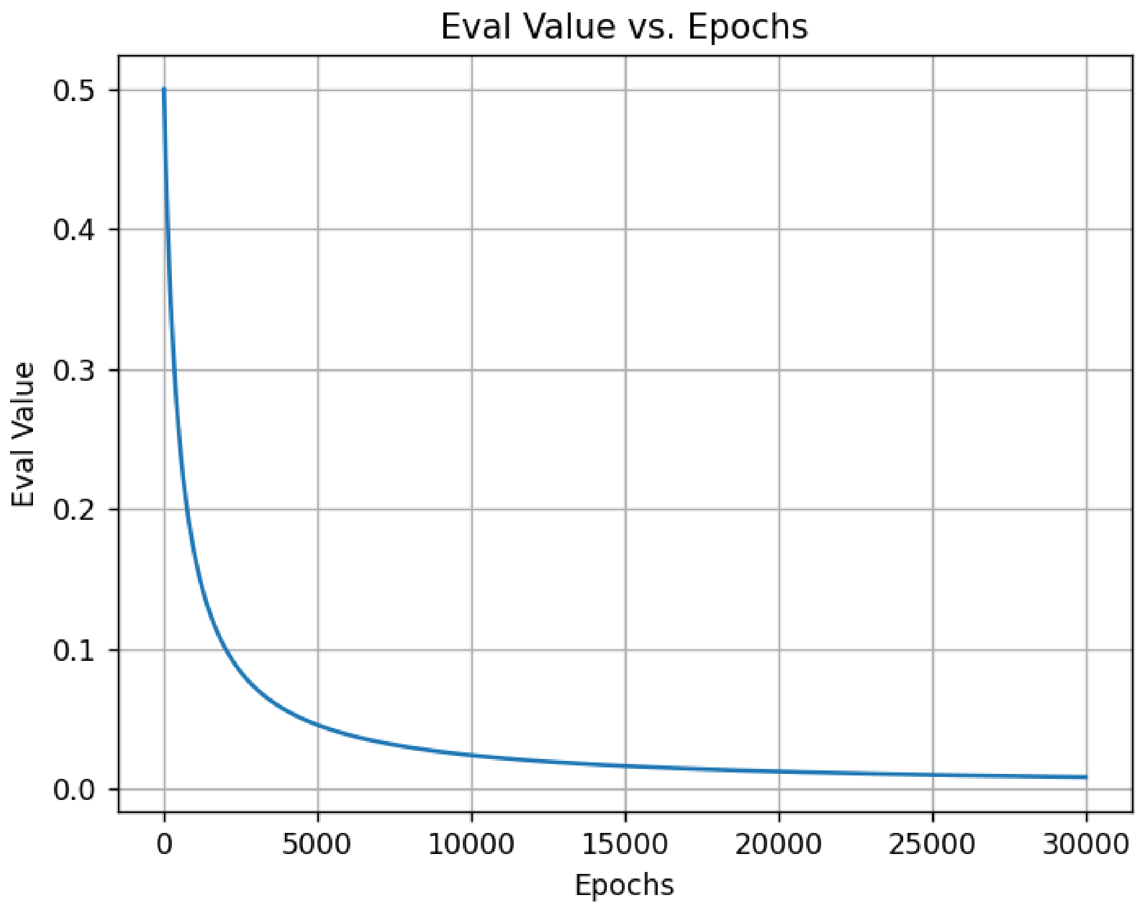
```
coefficients = compute_fourier_coefficients(p, x)
abs_square_coefficient_pos1 = torch.abs(coefficients[1])**2
abs_square_coefficient_neg1 = torch.abs(coefficients[-1])**2
print("coefficient_pos1:", abs_square_coefficient_pos1)
print("coefficient_neg1:", abs_square_coefficient_neg1)
print("energy_efficiency:", energy_efficiency(coefficients))

plt.plot(eval_values)
plt.xlabel('Epochs')
plt.ylabel('Eval Value')
plt.title('Eval Value vs. Epochs')
plt.grid(True)
plt.show()
```

Output and Plot

Optimization for Parallel Polarization Analyzer

Result:




```
Epoch: 27400 Eval: 0.008959934115409851
Epoch: 27500 Eval: 0.008927948772907257
Epoch: 27600 Eval: 0.008896077051758766
Epoch: 27700 Eval: 0.008864541538059711
Epoch: 27800 Eval: 0.008833317086100578
Epoch: 27900 Eval: 0.008802201598882675
Epoch: 28000 Eval: 0.00877120066434145
Epoch: 28100 Eval: 0.008740590885281563
Epoch: 28200 Eval: 0.008710218593478203
Epoch: 28300 Eval: 0.008679949678480625
Epoch: 28400 Eval: 0.008649789728224277
Epoch: 28500 Eval: 0.008620046079158783
Epoch: 28600 Eval: 0.008590501733124256
Epoch: 28700 Eval: 0.008561057969927788
Epoch: 28800 Eval: 0.008531716652214527
Epoch: 28900 Eval: 0.00850276742130518
Epoch: 29000 Eval: 0.008474026806652546
Epoch: 29100 Eval: 0.008445384912192822
Epoch: 29200 Eval: 0.008416841737926006
Epoch: 29300 Eval: 0.00838861707597971
Epoch: 29400 Eval: 0.0083606643602252
Epoch: 29500 Eval: 0.008332801051437855
Epoch: 29600 Eval: 0.008305035531520844
Epoch: 29700 Eval: 0.008277468383312225
Epoch: 29800 Eval: 0.008250277489423752
Epoch: 29900 Eval: 0.008223176002502441
Result: tensor([-3.1414+2.0552j, -3.1408+2.0552j, -3.1401+2.0552j,  ...,
         3.1401+2.0552j,  3.1408+2.0552j,  3.1414+2.0552j], requires_grad=True)
coefficient_pos1: tensor(0.0164, grad_fn=<PowBackward0>)
coefficient_neg1: tensor(1.2752e-09, grad_fn=<PowBackward0>)
energy_efficiency: tensor(1., grad_fn=<DivBackward0>)
```

 # eval function:

(1-α)(1-E)+α(|a_1|^2-|a_-1|^2)

where $$E = \frac{|a_1|^2 + |a_{-1}|^2}{\sum_{k=-\infty}^{\infty} |a_k|^2}$$

 # Theoretical Solution:

$$a_1 = a_{-1} = \frac{2}{\pi}$$

Reason may come from:

Improper weight factor of eval function, Using torch.fft.fft incorrectly, Error in grad descent, Bad initial value … …

Need Further Examination

Optimization for Parallel Polarization Analyzer

Follow-up Plan:

Optimization for Parallel Polarization Analyzer

- **1) 1D Scalar Grating**
  - Diffraction Grating
    - Traditional Diffraction Grating
    - No-gain Diffraction Grating
  - Phase-only Grating
    - Conflict Between x-space and k-space
    - Phase-only Grating Optimization
      - The Constrained Optimization Problem
      - The Least Squares Optimization Problem
      - The Minimum Variance Optimization Problem
    - Upper Bound on the Efficiency
      - The Generalized Constrained Optimization Problem
      - Numerical Results
  - Special Case
    - Non-trivial Analytical Solution: Two Beam Spliting
    - New Restrictions on Gratings: Binary Gratings
    - The Effect of Symmetry: Even Orders and Odd Orders Gratings
    - One Dimensional Continuous Gratings
- **2) 2D Scalar Grating**
  - Two Dimensional Fourier Series on General Lattices
  - Two Dimensional Symmetry
  - Two Dimensional Beam Splitting
    - The General Theory
    - Symmetry and Symmetry Breaking
    - Examples Using Square Gratings
      - Four Beam Splitting Using Square Gratings
      - Five Beam Splitting Using Square Gratings
      - Nine Beam Splitting With Square Gratings
      - Examples Using Hexagonal Gratings
      - Six Beam Splitting
      - Seven Beam Splitting with Hexagonal Gratings
  - Two Dimensional Continuous Gratings
- **3) 2D Matrix Grating**
  - Parallel Polarization Analysis by Matrix Gratings
    - Tetrahedron Grating
    - Cctahedron grating
  - Parallel Polarization Analysis by Linearly Birefringent Matrix Gratings
    - Linear Birefringence Implies $|p_k\rangle = |q_k^*\rangle$
    - Relation to Past Work on "Metasurface Polarization Optics"
    - A Unitary, Linearly Birefringent Matrix Grating for Polarization Analysis Can Only Have Two Orders Acting as Analyzers
  - Optimization-based Design Of Matrix Gratings For Parallel Polarization Analysis
    - Development Of An Optimization Scheme
      - Parameterization Of The Grating Unit Cell
      - Optimization
    - Results for 11-element Gratings
      - Progression of the Optimization
      - Optimized Results
    - Conversion to Metasurface Design

Red boxes: important examples I want to replicate

In Pytorch: parameters to be determined:

1D Tensor

2D Tensor

4 * 2D Tensor