Description

This notebook intends to gather all the functionalities you'll have to implement for assignment 2.2.

Load libraries

In [1]:
```
import numpy as np
import igl
import meshplot as mp
import time

import sys as _sys
_sys.path.append("../src")
from elasticsolid import *
from elasticenergy import *
from matplotlib import gridspec
import matplotlib.pyplot as plt

shadingOptions = {
    "flat":True,
    "wireframe":False,
}

rot = np.array(
    [[1, 0, 0 ],
     [0, 0, 1],
     [0, -1, 0 ]]
)
```

Load mesh

Several meshes are available for you to play with under `data/` : `ball.obj` , `dinosaur.obj` , and `beam.obj` . You can also uncomment the few commented lines below to manipulate a simple mesh made out of 2 tetrahedra.

In [2]:
```
v, _, _, t, _, _ = igl.read_obj("../data/beam.obj")
aabb = np.max(v, axis=0) - np.min(v, axis=0)
```

Linear/Non-Linear Elastic Solid

Instantiation

We first specify the elasticity model to use for the elastic solid, as well as pinned vertices, and volumetric forces.

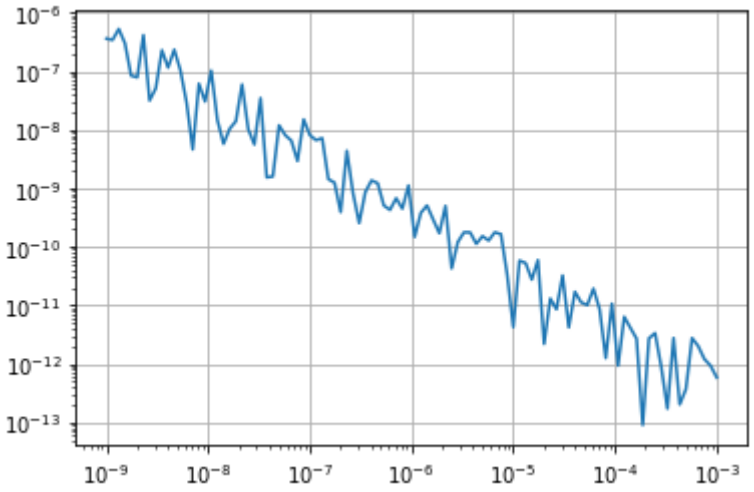
In [6]:

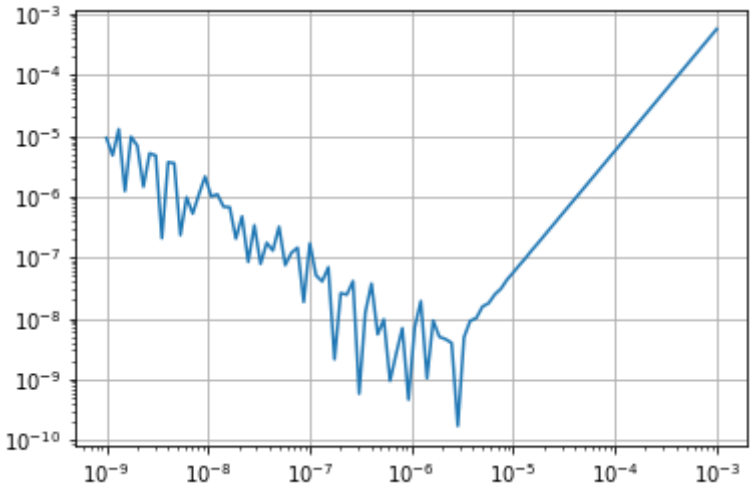
```
rho      = 131  # [kg.m-3]
young    = 10e6 # [Pa]
poisson = 0.2
force_mass = np.zeros(shape = (3,))
force_mass[2] = - rho * 9.81

minX     = np.min(v[:, 0])
pin_idx = np.arange(v.shape[0])[v[:, 0] < minX + 0.2*aabb[0]]

ee     = LinearElasticEnergy(young, poisson)
neo_ee     = NeoHookeanElasticEnergy(young, poisson)
solid = ElasticSolid(v, t, ee, rho=rho, pin_idx=pin_idx, f_mass=force_mass)
neo_solid = ElasticSolid(v, t, neo_ee, rho=rho, pin_idx=pin_idx, f_mass=for
```

Validate Elastic Energy Gradient (Elastic Forces) using Finite Differences on Elastic Energy

In [8]:

```
fd_validation_elastic(solid)
fd_validation_elastic(neo_solid)
```

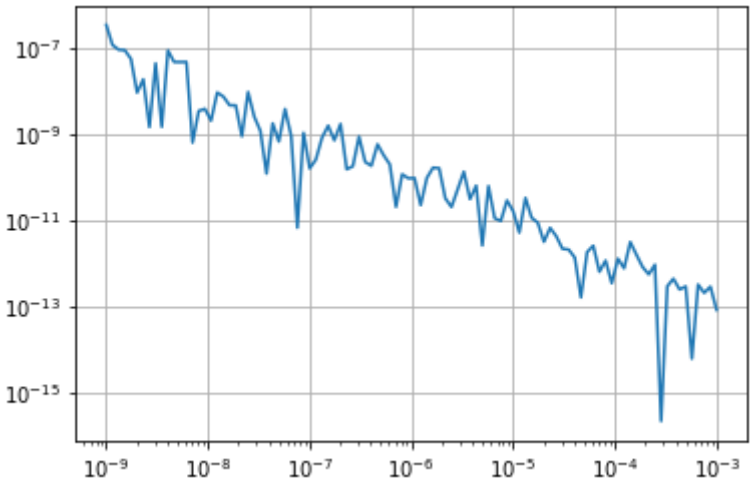


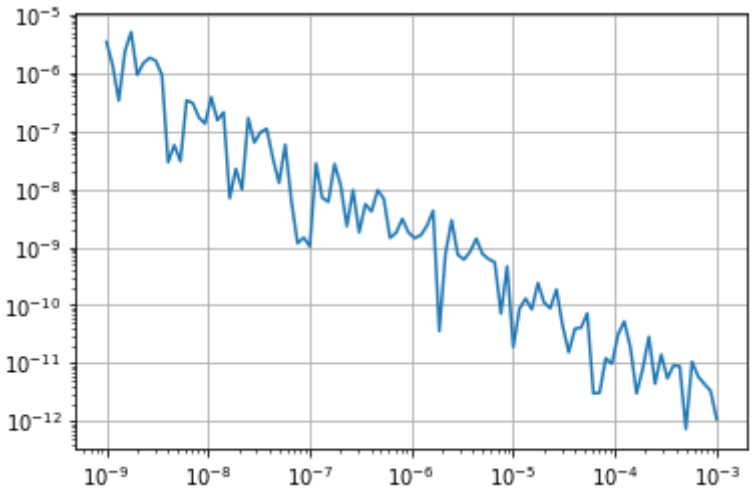


Validate External Energy Gradient (External

Forces) using Finite Differences on External Energy

In [9]:
```
fd_validation_ext(solid)
fd_validation_ext(neo_solid)
```





In [ ]:
```
```