SHAHID BEHESHTI UNIVERSITY

Deposition Models

Student name: *Hasti Hojabr - 97216040*

Date: *May 6, 2022*

Introduction

Two basic types of deposition models are random deposition model and ballistic deposition model. We studied power-law distributed noise and found the slope in log(t)-log(W) for both models.

$$W^2(t) = \frac{1}{L}\sum_{i=1}^{L}(h(i,t) - \frac{1}{L}\sum_{i=1}^{L}h(i,t))^2 \tag{1}$$

Scaling exponents α and β are slope of the plot after and before the cross over. We can see models and their scaling exponents in Fig1. Their values define the growth mechanism.

For random deposition the particles stick at the surface; for random deposition with surface diffusion, the particles do not stick irreversibly, but can relax to a nearby site of lower height. The ballistic deposition is different from the random deposition because the lateral sticking is allowed.

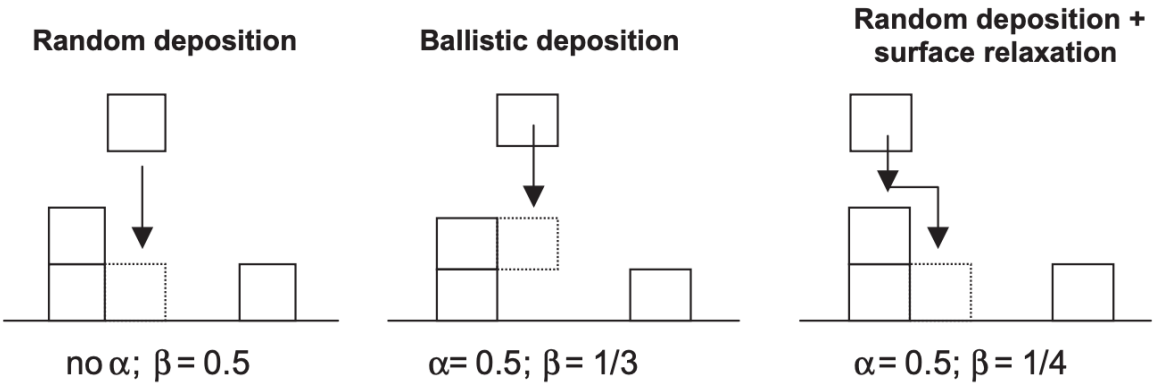

Figure 1: Discrete growth models and their scaling exponents α and β.

We haved used python3 with libraries such as numpy, matplotlib and scripy. For less computational time we used fortran90. The results are the same.

Random deposition model

For random deposition the particles fall randomly on the surface of lenght L and stick to it.

```
def RandomDeposition(L,T,E): #lenght , time , number of ensambles
    h,l = np.zeros(L),np.linspace(0,L,L)      ;   W=[]
    for e in range(E):
        h,l = np.zeros(L),np.linspace(0,L,L)  ; w,x,y = [],[],[]
        for t in range(T):
            i = int(np.random.uniform()*L+1)%L
            h[i] += 1
            if e==E-1: x.append(l[i]) ; y.append(h[i%L])
            w.append(np.std(h))
        W.append(w)
    return np.average(W, axis=0),np.arange(0,T,1),x,y,W
```

As we can see slope of log-log plot is 0.5. For five different lengths $L = 5, 10, 20, 40$ we can see width vs thickness in fig.2.

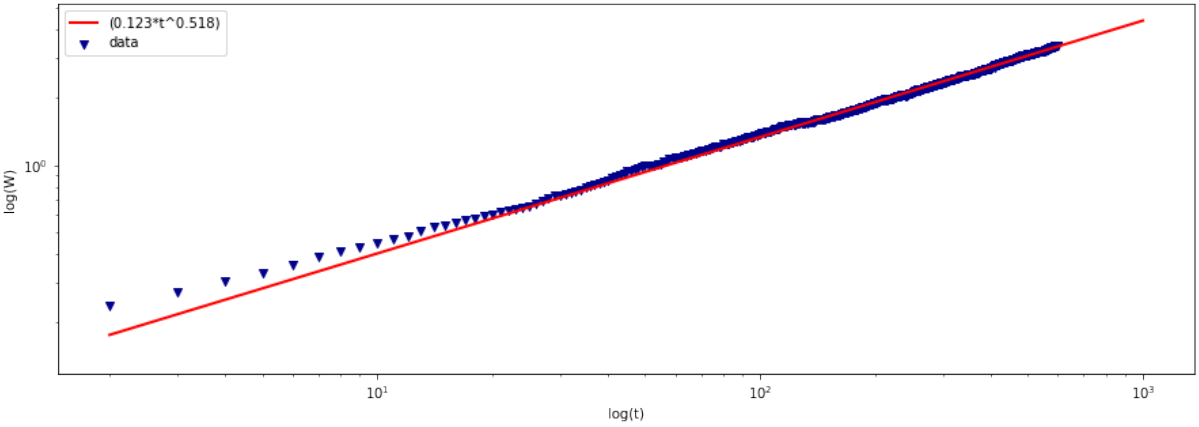

Figure 2: Power-law distributed noise for L = 20.

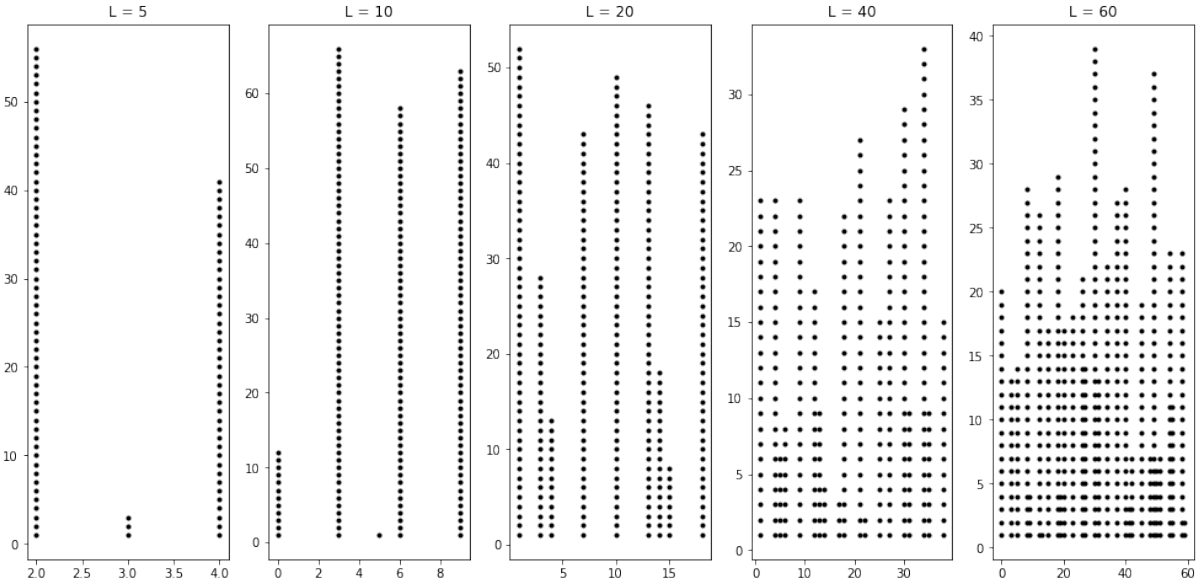

Figure 3: Width vs thickness for L = 5, 10, 20, 40 and 60.

Ballistic deposition model

Ballistic deposition model is a random deposition model with "sticky" blocks. Unlike random disposition, Ballistic deposition has a transition in t^*. After it noise does not grow with time and maintains a constant value.

```
def BallisticDeposition(L,T,E): #lenght , time , number of ensambles
    start=time.time()
    h,l = np.zeros(L),np.linspace(0,L,L)      ;   W=[]
    for e in range(E):
        h,l = np.zeros(L),np.linspace(0,L,L)  ; w,x,y = [],[],[]
        for t in range(T):
            i = round(np.random.random_sample()* L)
            h[i%L] = np.max([h[(i-1)%L],h[(i)%L],h[(i+1)%L]])+1
            if e==E-1:    x.append(l[(i+1)%L]) ;   y.append(h[i%L])
            w.append(np.std(h))
        W.append(w)
    time.sleep(1)
    end=time.time()
    print('Wall time:',end-start,'s.')
    return np.average(W, axis=0),np.arange(0,T,1),x,y,W,end-start
```

Function will give us average interface width over number of ensembles, time, width, thickness, a list of all interface widths and wall time. Wall time increase with lattice size. It is measured in seconds.

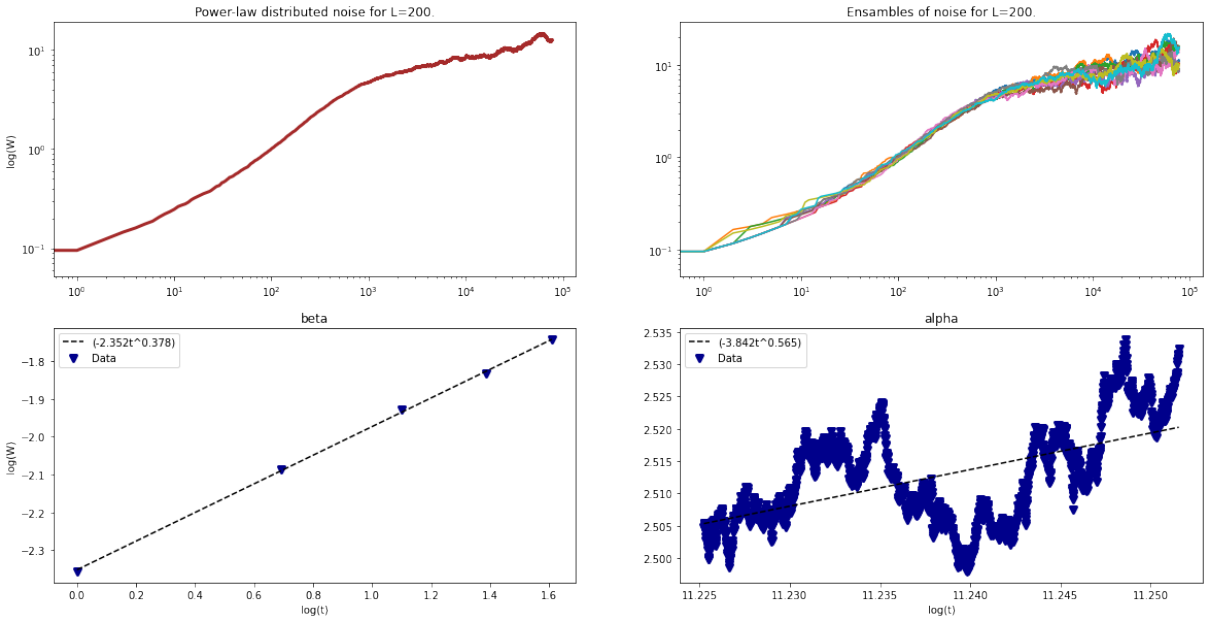

Figure 4: Power-law distributed noise for surface length of 220.

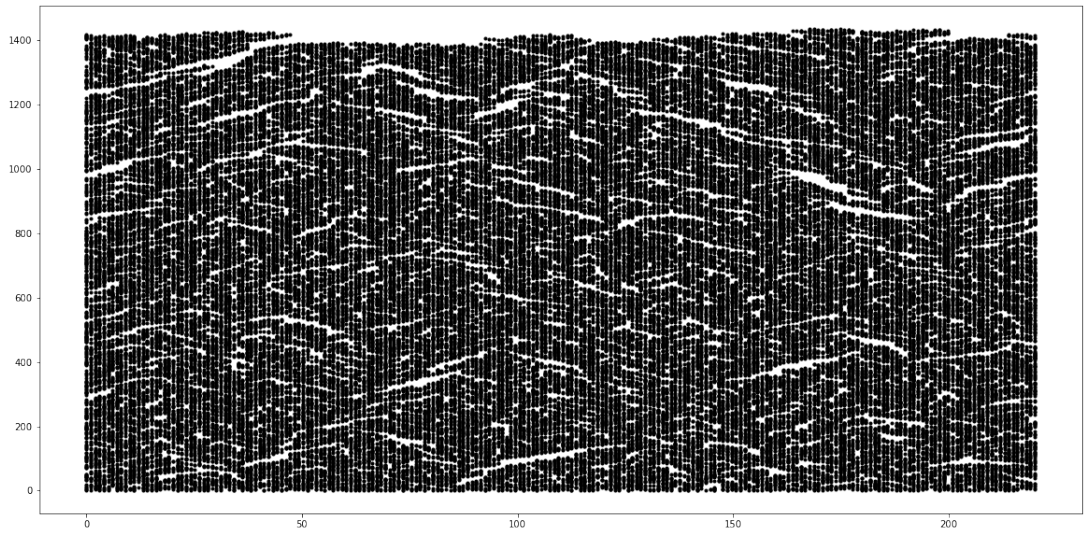

Figure 5: Width vs thickness for surface length of 220.

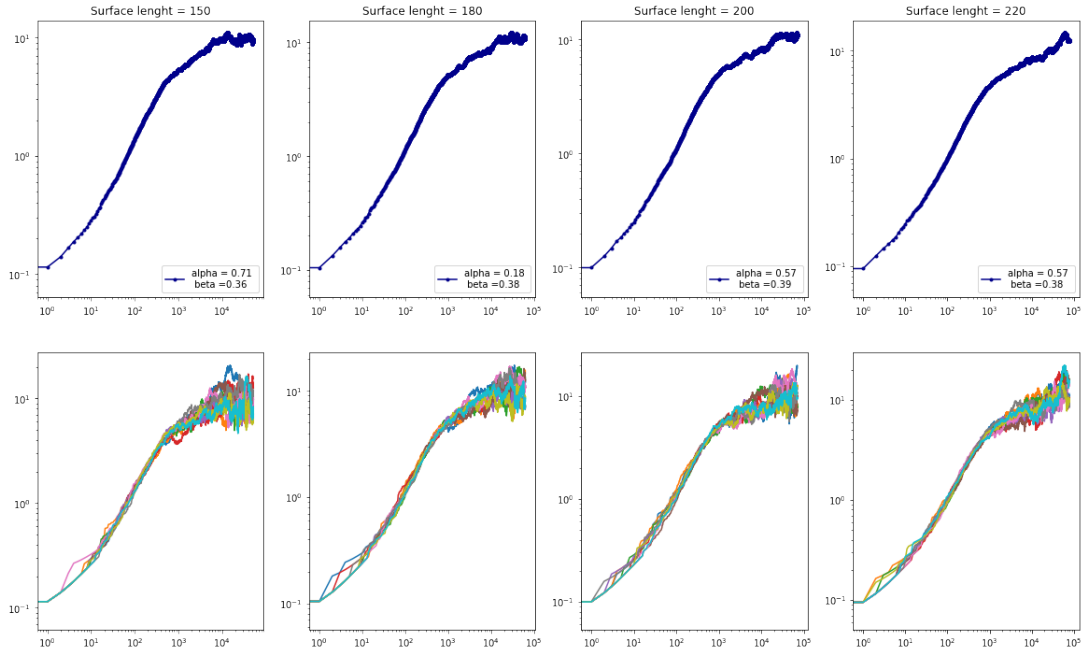

Figure 6: Power-law distributed noise for surface length of 150, 180, 200 and 220.

Wall time for different length can be approximated using a linear fit. It will give us

$$t_{wall}(L,t) = 0.25L - 1.13 \tag{2}$$

So approximately we could run a system with L=1000 on a local computer with python3 and it would take 40 minuets.

With right value of scaling exponent we could use the scaling relation below. All the plots will collapse into the same line and length-dependent effects disappear.

$$w(L,t) = L^{\alpha} f(t/l^{z}) \tag{3}$$

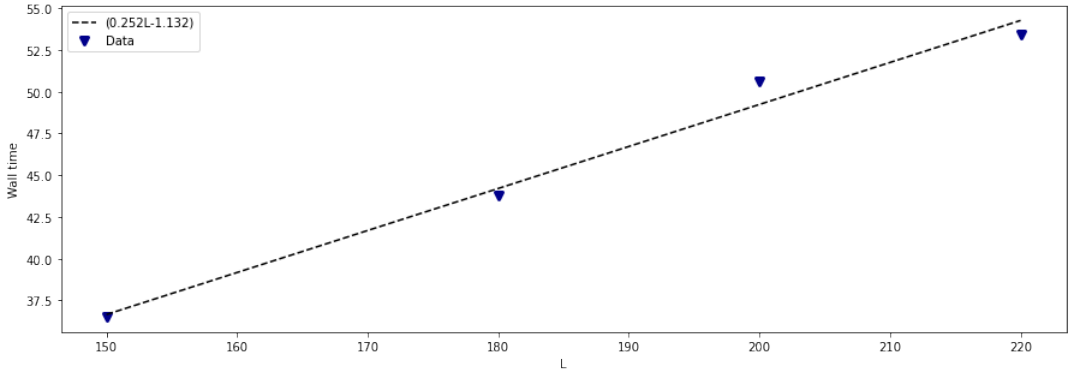

Figure 7: Lenght vs wall time.

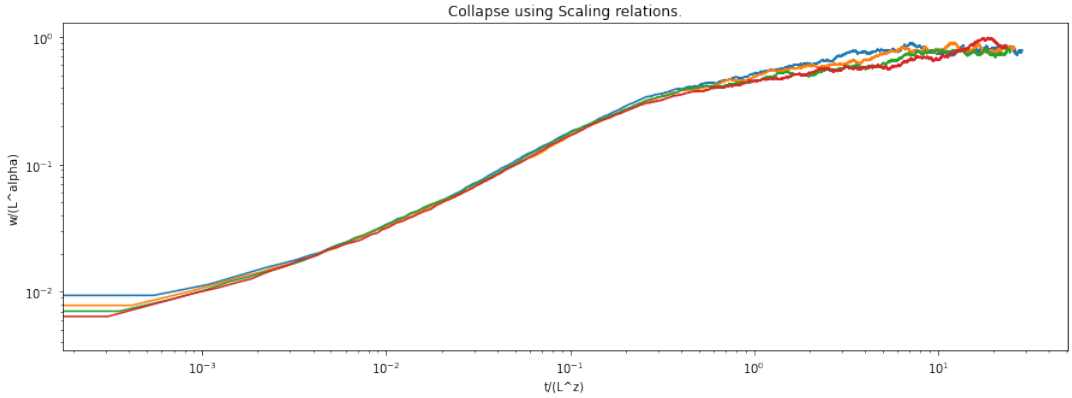

Figure 8: Lenght vs wall time.

Random deposition model with surface relaxation

For induction of surface relaxation in RD model we allow particle to diffuse along surface and choice the lowest height for its position. We studied power-law distributed noise and found the slope in log(t)-log(W) for this model.

```
def RandomDepositionRelaxed(L,T,E): #lenght , time , number of ensambles
    start=time.time()
    h,l = np.zeros(L),np.linspace(0,L,L)      ;   W=[]
    for e in range(E):
        h,l = np.zeros(L),np.linspace(0,L,L)  ; w,x,y = [],[],[]
        for t in range(T):
            i = round(np.random.random_sample()* L)
            minn = min([h[(i-1)%L],h[(i)%L],h[(i+1)%L]])
            if h[(i-1)%L]==minn : h[(i-1)%L]+=1
            elif h[(i)%L]==minn : h[(i)%L]  +=1
            elif h[(i+1)%L]==h[(i-1)%L]==minn:
                if np.random.rand()<0.5: h[(i+1)%L]+=1
                else: h[(i-1)%L]+=1
            else                : h[(i-1)%L]+=1
            w.append(np.std(h))
            if e==E-1:    x.append(l[(i+1)%L]) ;   y.append(h[index])
```

```
        W.append(w)
    time.sleep(1)
    end=time.time()
    print('Wall time:',end-start,'s.')
    return np.average(W, axis=0),np.arange(0,T,1),x,y,W,end-start
```

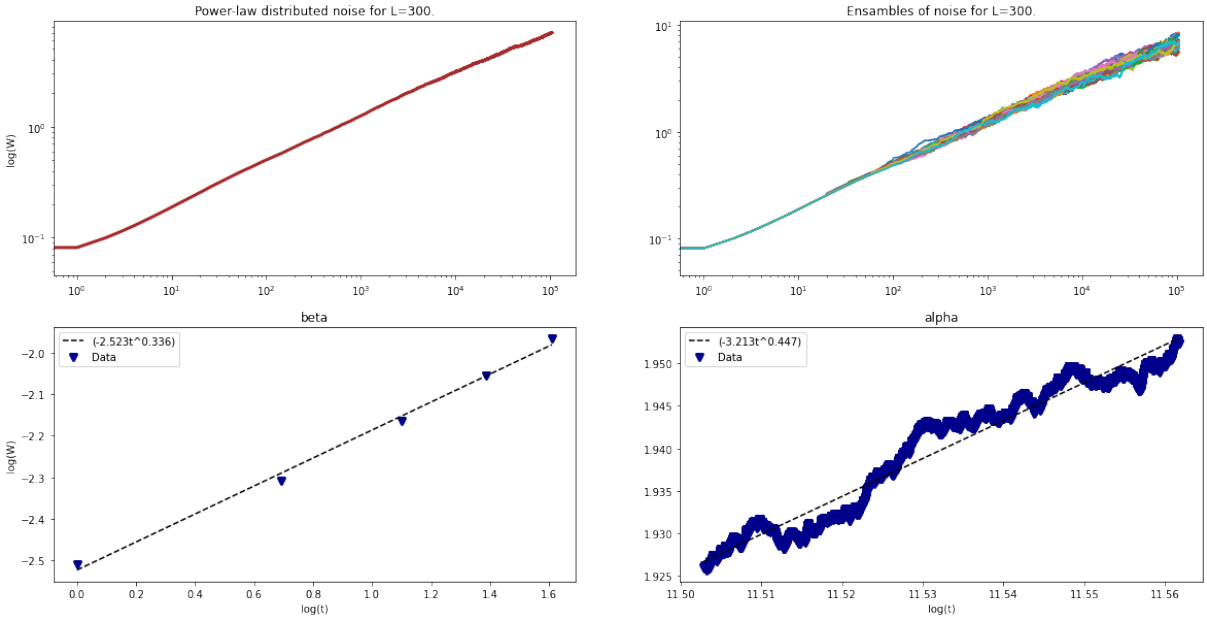

Figure 9: Power-law distributed noise for surface length of 300.

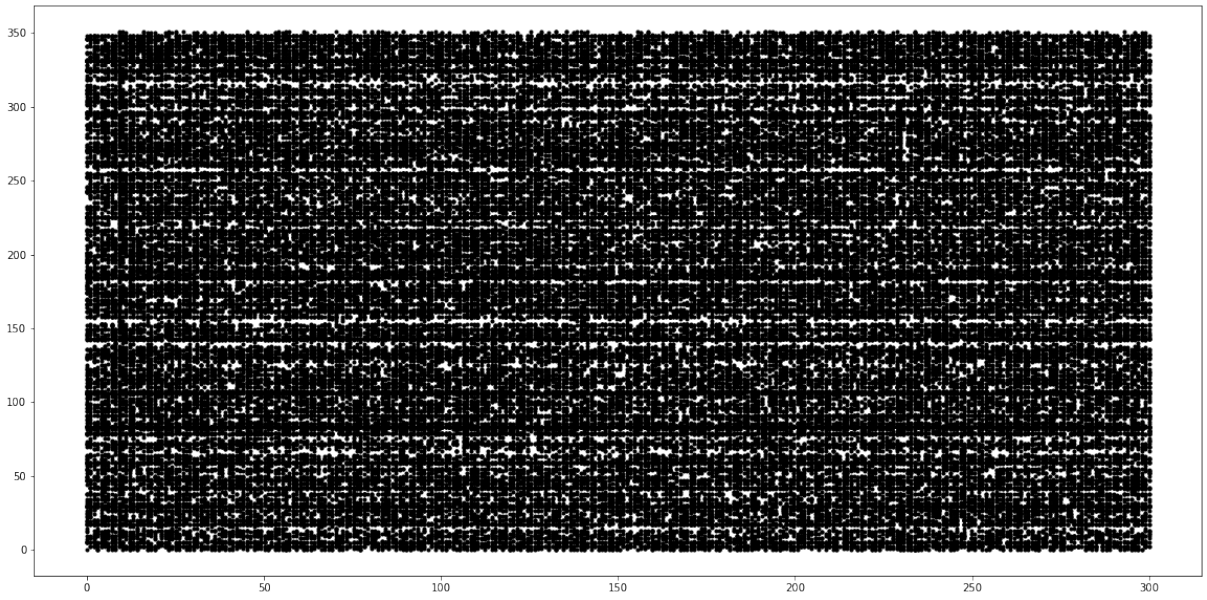

Figure 10: Width vs thickness surface length of 300.

Code

We have used python3 in Jupyter notebooks for this homework. Notebook can be downloaded from link below.

Google Collab - Deposition models