Supporting Information

to accompany

Measuring Local Atomic Structure Variations Through the Depth of Ultrathin (<20 nm) ALD Aluminum Oxide: Implications for Lithium-Ion Batteries

Nikhila C. Paranamana[1], Ryan Gettler[2], Henry Koenig[1], Stephen Montgomery-Smith[3], Xiaoqing He[4,5], Matthias J. Young[1,2,*]

[1]Department of Chemistry, University of Missouri, Columbia
[2]Chemical Engineering, University of Missouri, Columbia
[3]Department of Mathematics, University of Missouri, Columbia
[4]Electron Microscopy Core, University of Missouri, Columbia
[5]Mechanical and Aerospace Engineering, University of Missouri, Columbia

A: **Structure files derived from ALD AlO_x at Positions 1, 2, and 3**

The .xyz files used to calculate structural metrics at 1st, 2nd & 3rd positions of AlO_x coating as described in the main text can be found as separate attachments in the supporting information.

B: **Mathematical and computational calculation of electron beam intersecting volume**

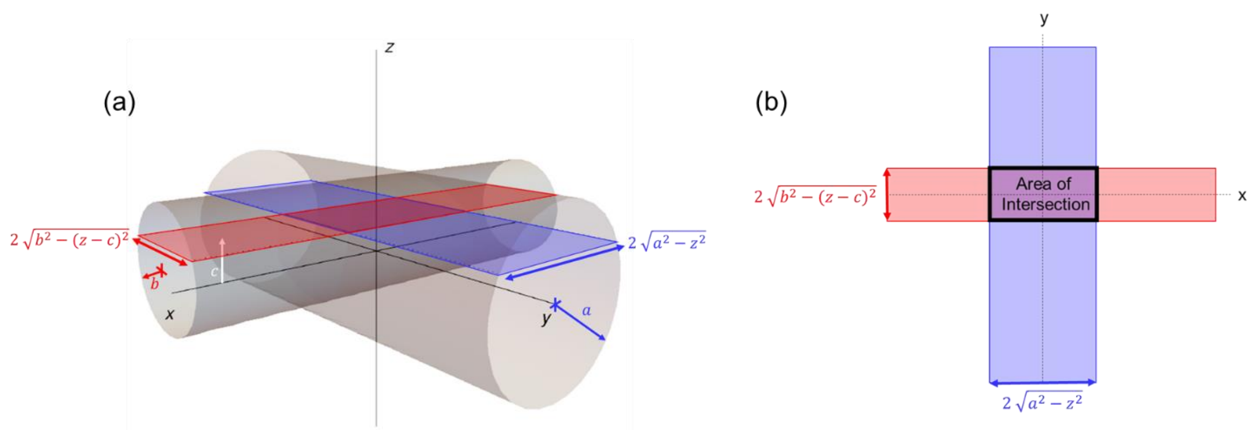

Figure S1. (a) 3D depiction of the intersection of two perpendicular cylinders of radius a and b, offset by a distance c and the corresponding (b) area of intersection in the x-y plane at a fixed height z, yielding a simple expression to calculate the intersecting volume of the two cylinders.

Assuming there are two infinite cylinders, one of radius a, and the other of radius b. Also, assuming that the axes of the cylinders are in directions perpendicular to one another, and that the shortest distance between these two axes is c. Then the volume of the intersection of the cylinders is given by

$$V = 4\int_{\max\{-a,-b+c\}}^{\min\{a,b+c\}} \sqrt{a^2-z^2}\sqrt{b^2-(z-c)^2}\,dz.$$

An analytic formula for this integral involves elliptic integrals and is complicated. But it is very easy to compute this integral numerically. To see this, without loss of generality, let us suppose that the equations of the two cylinders are,

$$C_1 : x^2 + z^2 \leq a^2,$$

$$C_2 : y^2 + (z - c)^2 \leq b^2.$$

Let us consider a layer of the intersection of these two cylinders at a particular value of z. Then we see that this layer is non-empty if and only if,

$$|z| \leq a, \;\; |z - c| \leq b,$$

and that this layer is the rectangle given by,

$$|x| \leq \sqrt{a^2 - z^2}, \qquad |y| \leq \sqrt{b^2 - (z - c)^2}.$$

Now one simply applies the formula for the area of a rectangle. The python code to evaluate this formula is given below using the python function `volume_intersecting_cylinders`.

Below, we also provide the python code for the specific use case of calculating what fraction of the electron diffraction signal is arising from each of three cylindrical shells in a multilayer stack as manifested in the main text for localized electron diffraction at varying positions along the depth of a film coated on a CNT substrate, here an AlO_x ALD film comprised of an outer surface, bulk AlO_x, and inner surface (CNT interface) structure.

Example Usage:

```
ipython
import intersect_cylinders.py import ED_scan
ED_scan(10,2.5,0.5,9,2.5)
```

```python
#filename: intersect_cylinders.py
import math
import scipy.integrate
import pandas as pd

def ED_scan(a=10,b=2.5,h1=0.5,h2=9,h3=2.5,filename='output.xlsx'):
    """
    Returns fraction of electron diffraction signal arising from each of
three coating layers on a cylindrical substrate as a function of distance
from the center of the cylindrical substrate
    a = radius of cylindrical substrate in nm
    b = radius of diffracting electron beam in nm
    h1 = thickness of outer coating layer 1 in nm
    h2 = thickness of middle coating layer 2 in nm
    h3 = thickness of inner coating layer 3 in nm
    """
    ht = h1+h2+h3; #total coating thickness

    num_points=int((ht-b)*100); #equal spacing over thickness
    c=[0]*num_points;
    f1=[0]*num_points;
    f2=[0]*num_points;
    f3=[0]*num_points;
    for i in range(0,num_points):
        c[i]=ht+a-float(i)/100;
        V1= volume_intersecting_cylinders(ht+a,b,c[i]);
        V2= volume_intersecting_cylinders(ht+a-h1,b,c[i]);
        V3= volume_intersecting_cylinders(a+h3,b,c[i]);
        f1[i]=(V1-V2)/V1; #fraction of intersecting volume from layer 1
        f2[i]=(V2-V3)/V1; #fraction of intersecting volume from layer 2
        f3[i]=(V3)/V1; #fraction of intersecting volume from layer 3

    df = pd.DataFrame({'c':c, 'f1':f1, 'f2':f2, 'f3':f3})
    df.to_excel(filename,index=False)
    return df

def volume_intersecting_cylinders(a,b,c,verbose = False):
    """
    Returns volume of intersection of two perpendicular cylinders.
    a = radius of one cylinder,
    b = radius of the other cylinder,
    c = distance between their axes.
    verbose: do you want to print out upper bound of absolute error?
    """

    lower = max(-a, c-b);
    upper = min(a, c+b)
    if upper < lower:
        return 0
    else:
        integral =scipy.integrate.quad(lambda x: 4 * math.sqrt(a**2 - x**2)*\
        math.sqrt(b**2 - (x-c)**2), lower, upper)
        if verbose:
            print("Error is at most",integral[1]);
        return integral[0]
```