**Precision warning!**

def interval\_t(size,num\_val=50,kmax=None):  
 ### Generate sequence of interval times, k  
 if kmax is None:  
 k\_stop = size//2  
 else:  
 k\_stop = kmax  
 if k\_stop > size//2:## prohibit going larger than N/2  
 k\_stop = size//2  
 print("Warning: k cannot be longer than N/2")  
 #print(size)  
 #print(size // 2)  
 #print(np.log2(k\_stop))  
 np.set\_printoptions(precision=30)  
  
 #print(np.longdouble(math.log2(np.longdouble(k\_stop))))  
 k = np.logspace(start=np.log2(2),stop=np.log2(k\_stop), endpoint=True, base=2,num=num\_val,dtype=int)  
  
  
 ####################################################################  
 # WARNING !!!  
 ####################################################################  
 k[len(k) - 1] = k\_stop  
  
  
 #k1 = np.around(k, decimals=14)  
 #print(k1)  
 #k = k.astype(dtype=np.int, copy=False)  
 #y = np.linspace(np.log2(2), np.log2(k\_stop), num=num\_val, endpoint=True)  
 #y = np.power(2,y)  
 #print(y)  
 #y[1] = 100.  
 #getcontext().prec = 28  
 #print(Decimal(math.log2(Decimal(k\_stop))))  
  
 #y = np.linspace(Decimal(np.log2(2)), Decimal(math.log2(Decimal(k\_stop))), endpoint=True, num=num\_val, axis=0)  
  
  
  
  
 #print(y)  
 #if dtype is None:  
 # return \_nx.power(base, y)  
 #return \_nx.power(base, y).astype(dtype, copy=False)  
  
  
  
  
  
  
 #print(k)  
  
  
  
  
  
 # !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! OWN CODE !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!  
 # Because bug in np.logspace with dtype = np.float?  
 # 218 - 108 integer; 220 - 110 integer;  
 #print(y)  
 #y = y.astype(dtype=np.int, copy=False)  
 #print(y)  
  
  
 return np.unique(k);