Algorithm 1: Expanding Adaptive Threshold

Input: Input
v_{in} - the volume to operate on
n_x - x neighborhood to filter over
n_y - y neighborhood to filter over
n_z - z neighborhood to filter over
p - the percentile for the adaptive threshold
minSigma - the minimum standard deviation of the neighborhood
Output: Output
v_{out} - The filtered volume

subXLen = $v_{in}.shape[0]/s_x$
subYLen = $v_{in}.shape[1]/s_y$
subZLen = $v_{in}.shape[2]/s_z$
for $xInc/in1, ..., s$ **do**
  for $yInc/in1, ..., s$ **do**
    for $zInc/in1, ..., s$ **do**
      origSubVolume = v_{in} centered at xInc, yInc, zInc with neighborhood x+-subXLen/2, y+-subYLen/2, z+-subZLen/2
      subVolume = origSubVolume
      a = 0
      curSigma = stdDev(subVolume)
      while $curSigma < minSigma$ **do**
        a+=1
        subVolume = v_{in} centered at xInc, yInc, zInc with neighborhood x+-subXLen/2 + a, y+-subYLen/2 + a, z+-subZLen/2 + a
        curSigma = stdDev(subVolume)
      subThresh = binaryThreshold(subVol, p)
      v_{out}[origSubVolume] = subThresh[origSubVolume]
return v_{out}