Measurement of Average Aggregate Density by Sedimentation and Brownian Motion Analysis

Richard E. Cavicchi[1], Jason King[1,2], and Dean C. Ripple[1]
[1]Bioprocess Measurements Group
National Institute of Standards and Technology, Gaithersburg, MD 20899
[2]now with XSOLIS, 301 Plus Park Blvd. Nashville, TN 37217 Suite 411

ABSTRACT
The spatially averaged density of protein aggregates is an important parameter that can be used to relate size distributions measured by orthogonal methods, to characterize protein particles, and perhaps to estimate the amount of protein in aggregate form in a sample. We obtained a series of images of protein aggregates exhibiting Brownian diffusion while settling under the influence of gravity in a sealed capillary. The aggregates were formed by stir-stressing a monoclonal antibody (NISTmAb). Image processing yielded particle tracks, which were then examined to determine settling velocity and hydrodynamic diameter down to 1 μm based on mean square displacement (MSD) analysis. Measurements on polystyrene calibration microspheres ranging in size from 1 μm to 5 μm showed that the MSD diameter had improved accuracy over the diameter derived from imaged particle area, suggesting a future method for correcting size distributions based on imaging. Stokes' law was used to estimate the density of each particle. It was found that the aggregates were highly porous with density decreasing from 1.080 g/cm^3 to 1.028 g/cm^3 as the size increased from 1.37 μm to 4.9 μm.

Keywords: Flow imaging, Image analysis, Microscopy, Particle Sizing, Physical Characterization, Protein aggregates,

INTRODUCTION

The use of orthogonal methods to characterize the size distribution of protein aggregates in biopharmaceuticals is useful to cover broader size ranges with increased confidence. However, it can be difficult to reconcile results obtained through different measurement methods, which—based on their respective calibration and operational protocols—may provide seemingly conflicting results. The spatially averaged density is a key parameter for comparing size distributions obtained by the resonance mass method, particle tracking, flow image analysis, or light obscuration. Extracting particle diameter from a resonance mass measurement requires knowledge of the particle density. Optical methods also depend on particle density in an indirect manner: a higher particle density corresponds to a higher optical scattering or image contrast, which in turn can affect the particle diameter measurement. The density of protein aggregates is also important for estimating the amount of aggregated protein in a sample.

The density of a protein (as a molecule or pure substance) has long been considered a parameter that is—to a good approximation—independent of the protein type, and is commonly used in the analysis of X-ray structure data from protein crystals. A protein density of 1.35 g/cm^3 is a commonly accepted value based on early sedimentation[1] and compressiblity[2] measurements over a variety of different proteins)[3]. Quillin and Matthews[3]

revised the calculation of Andersson and Hovmoller[4] of the molecular volume based on the crystal structures for 28 proteins using coordinates from the Protein Data Bank[5] to estimate an average density of 1.43 g/cm^3. Fischer et al. reanalyzed the results obtained by Quillin and Matthews (2000), noting a dependence of density on molecular weight and suggesting proteins >30 kDa have a 1.41 g/cm^3 density value[6]. An example application for the protein density value is given by Fischer et al.[7], who used small angle X-ray scattering to determine protein volume. A protein density value based on the results from Squire et al.[1] and Gekko et al.[2] (1.37 g/cm^3) was then used to determine molecular weight.

There is evidence that protein aggregates are porous. Sung et al. found that "TEM revealed that aggregates were highly irregular in shape and porous in nature suggesting that their water content is substantial[8]." This would suggest an average density lower than the value for pure protein. There are several reports of researchers using aggregate density to estimate the amount of protein in an aggregate based on imaging results. Wuchner et al. assumed a value of 1.3 g/cm^3 for the density. They note the uncertainty resulting from such an assumption: "By applying a hard sphere model to calculate the weight percent of protein particles (an approximation since the protein particles are heterogeneous in nature), these calculations may be an overestimate of their mass[9]." Barnard et al. [10] combined the pure protein density value from Quillin and Matthews[3] (1.43 g/cm^3) with an estimate that aggregates are 25 % by volume water, resulting in an estimated aggregate density of 1.32 g/cm^3. Kalonia et al.[11] combined the pure protein value from Fischer et al. [6] (1.41 g/cm^3) with an estimate that aggregates are 80 % by volume water, resulting in an estimated aggregate density of 1.08 g/cm^3.

Recently, Folzer et al. sought to measure protein aggregate densities for the purpose of converting protein aggregate resonance mass measurements to an equivalent spherical diameter[12]. Using cesium chloride solutions of varying concentrations, they measured the buoyant mass of particles using the resonance mass method as a function of solution density. It was assumed that the particle density is equal to the solution density where the linear fit of the data passes through zero buoyant mass. This procedure yielded excellent values for the density of calibration microspheres. For protein aggregates, however, it neglects the possibility that the particles may be porous, and that the altered density fluid—as it penetrates the pores of the particle—changes the total mass of the aggregate. The resulting density from this measurement is actually (Mass)/(Non-Buffer-Accessible Volume), and not the average density of the particle (Mass)/(Total Volume) in the formulation buffer. The measured values ranging from 1.28 g/cm^3 to 1.33 g/cm^3 correspond to the density of the non-buffer-accessible portion of the particle, which is closer to the value for pure protein.

In this work, we seek to measure the sedimentation velocity of protein aggregates settling in solution under the influence of gravity while simultaneously measuring the particle size. Similar measurements have been reported for other types of particles. Bach et al. performed sinking velocity measurements of various phytoplankton species using a flow imaging system (FlowCam) outfitted with glass cuvettes[13]. The equivalent spherical diameter (ESD) was obtained from the instrument's software determination of the area of imaged particles, while the velocity was obtained by tracking the particles using a MATLAB script. Sediq et

al. used a FlowCam to perform sedimentation and size measurements on Poly lactic-co-glycolic acid (PLGA) particles and polystyrene microspheres ranging in size from 30 μm to 70 μm[14]. PLGA is a biodegradable copolymer used for drug delivery or biomaterial applications. Falling velocity measurements in fluids of different densities were used in combination with the imaged size to determine particle density. The porous PLGA particles were in dried form prior to being added to the fluid, so the pores of the particles were assumed to be filled with air. Average densities of 0.8 g/cm^3, 1.0 g/cm^3, and 1.3 g/cm^3 were observed for three different batches of the PLGA. Zhao et al. measured the density of yeast cells by observed the falling velocity of cells from a top view and using an optical electrokinetics platform[15]. Cells on a bottom photoconducting electrode were propelled upward towards an upper transparent conducting electrode when an illumination source was activated. Switching off the illumination allowed the cells to sediment under gravity. By comparing the out-of-focus images obtained during sedimentation to calibrated images prepared prior to the light pulse, the vertical trajectory was obtained. Estimated densities for the cells ranged from 1.04 g/cm^3 to 1.13 g/cm^3.

In the measurements reported here, we use a configuration similar to Bach et al. [13] and Sediq et al. [14]. The aggregates—formed by stir-stressing NISTmAb— primarily produced particles below 5 μm in diameter. In this size range, size measurements based on particle image are prone to diffraction and out-of-focus effects. Instead, we used the tracked diffusive motion of the particle to obtain the equivalent spherical hydrodynamic diameter. By simultaneously imaging the particles, we were able to obtain a direct comparison between the image sizing and Brownian motion sizing methods for our experimental arrangement. Combined with the measurement of the average sedimentation velocity, we obtain the spatially averaged density difference between the liquid and settling particle. This density difference is equal to the buoyant mass (mass of protein material plus trapped liquid minus mass of displaced liquid) divided by the hydrodynamic volume. The average particle density is the sum of this measured density difference and the liquid density.

MATERIALS AND METHODS

Materials

Polystyrene microspheres nominally 1.0 μm, 2.0 μm, 3.0 μm, and 5.0 μm in diameter were obtained in aqueous suspensions from the line of Thermo Scientific™ Duke Standards™ 2000 Series Uniform Polymer Particles (ThermoFisher, Grand Island, NY). (Certain commercial products and instruments are identified to adequately describe the experimental procedure in this paper. In no case does such identification imply endorsement by the National Institute of Standards and Technology.) The microspheres were diluted in ultrafiltered, deionized water (Barnstead Nanopure system, Dubuque, IA). The water density was measured using a Laboratory Density Meter Model DDM 2911 (Rudolph Research Analytical, Hackettstown, NJ). The value was 0.997051 g/cm^3 at 25 °C with a standard deviation of 0.000002 g/cm^3. To consider if there were any concentration-dependent effects, samples were made with concentrations ranging from 1x10^6 ml^{-1} to 33x10^6 ml^{-1}.

The NISTmAb is a monoclonal antibody available as reference material (RM 8671) from NIST[16]. A vial of RM 8671 contains 800 µL of 10 mg/mL IgG1κ (molecular mass ≈150 kDa), a monoclonal antibody in histidine buffer: 12.5 mmol/L L-histidine and 12.5 mmol/L L-histidine HCl (pH 6.0). The stock NISTmAb was diluted to 1 mg/mL in histidine buffer and stressed by stirring for two hours as described previously by Joubert et al.[17] to create aggregates. The buffer density was measured to a value of 0.998817 g/cm^3 at 25 °C with a standard deviation of 0.000039 g/cm^3.

Experimental Methods

A schematic of the imaging setup is shown in Figure 1 a. Samples were drawn into rectangular-cross-section glass capillaries (VitroCom, Mountain Lakes, NJ) by capillary action. The glass capillaries have inner dimensions of 0.05 mm x 0.5 mm x 50 mm, and are sealed at both ends with silicone high vacuum grease. This sealing substance (Dow Corning Corporation, Auburn MI 48611) is water insoluble, is commonly used for sealing coverslips for microscopy applications, and had the advantage that imaging could begin immediately after sealing, which was important for protein aggregates that eventually would adhere to the capillary walls. No evidence of leaching of the sealant into the tube was observed, and the results were similar to other sealing methods we also evaluated (Room-Temperature-Vulcanizing silicone glue, epoxy). A flow imaging system (FlowCam, Fluid Imaging Technologies Inc., Yarmouth, ME) was modified to replace the flow-through system with a glass capillary. The use of the glass capillary was critical to ensure there was no convective fluid flow during the long settling measurements. The FlowCam used for these experiments features an open component layout; the lid on the instrument was closed during measurements to mitigate thermal gradients that could cause convection. Subsequent image sorting along the x-axis revealed no evidence of convection or non-uniformity. The temperature was measured during each run with a type K thermocouple (Model DP462, Omega Eng. Inc., Norwalk, CT) and was used to obtain the viscosity based on the temperature dependence of the viscosity of water[18]. Images were collected as the particles settled in the glass capillary. The system was configured with 20× magnification using a lens that was modified by temporarily gluing a 2.5 mm aperture to the rear of the objective to increase the depth of field. This modification reduced the possibility of particles drifting out of focus during long tracking measurements. The use of a 20× objective also results in greater spatial resolution (compared to a 10× objective), and thus more precise MSD and settling velocity values. The images were 1294 pixels x 964 pixels (362 µm x 270 µm.) Time-stamped images were typically acquired at a rate of 1 frame per second. It was useful to acquire runs of 2000 s or longer to obtain tracks that have a falling velocity displacement that represents a significant component of the total vertical displacement when compared to the Brownian diffusion contribution. The low acquisition speed reduced the obtained quantity of data to a manageable level. There was little wall adhesion during measurements of the calibration microspheres, allowing runs as long as 16000 s to analyze a larger number of particles. However, the protein aggregate samples exhibited significant adhesion to the capillary walls after about 2000 s, therefore, multiple

runs with freshly loaded capillaries were studied. Reported here are the results from 17 separate NISTmAb-loaded capillaries.

Images were analyzed using the NIST "Variable Threshold" algorithm, which improves boundary definition for samples with particles of widely varying contrast[19]. An example of some processed images is shown in Figure 1 b. The software implementing this algorithm, FIJI[20], is a freely available derivative of ImageJ[21] preloaded with additional plugins. The software removes the background by taking the median of a set of images and subtracting that median image from each image[19]. The software produces a table of data that includes particle center of mass coordinates, radius, and mean brightness. The results were converted by a program (Supplementary Information) to a file in a format usable by Trackmate[22]—an open source plug-in for FIJI—which was used to identify particle tracks. Trackmate was modified to export data that included particle radius and mean brightness (Supplementary Information). Tracks ranged in length up to 6986 steps (1 s per step) for single 1 μm microspheres. Tracks less than 50 steps were removed from analysis. Tracks in which the particle is adhered to the capillary walls at the end of the track were also removed from analysis, as these tracks might be affected by wall effects that would modify calculated properties. For the microspheres, size and circularity filtering of the tracks was performed so that only monomers of the analyzed spheres were analyzed. Filtering of tracks was not employed for the protein aggregate samples. When one track intersected another track, both tracks were deemed ended. The total number of tracks analyzed for both microspheres and aggregates are shown in Table 1.

Track Analysis

a. Determination of hydrodynamic diameter from Brownian motion

The analysis to determine particle size is similar to what is used in nanoparticle tracking analysis (NTA); the main difference is that the particle's coordinates are based on the particle image, rather than the light scattering distribution. Tracks were analyzed using a mean-square-displacement (MSD) algorithm to calculate a hydrodynamic diameter[23-25] and a settling velocity. The program is supplied in the Supplementary Information. For each particle, the MSD was calculated over the first *n* time intervals between measurements[25]. For this work, *n* was chosen as 10, which gave good results for particle diameters ranging from 1 μm to 10 μm. For a two-dimensional diffusion process, the MSD scales according to:

$$MSD(\tau) = 4D\tau \,, \qquad (1)$$

where D is the diffusion coefficient and τ is the time interval between measurements. MSD is calculated by the following:

$$MSD(t) = \langle [r(t) - r(0)]^2 \rangle \qquad (2)$$

where *r* is the two-dimensional position of the particle and *t* is the timestamp for the position measurement. A linear fit to the results from Equation 2 is used to extract the

diffusion coefficient, D, which can then be used to determine diameter, d, using the Stokes-Einstein equation

$$d = \frac{kT}{3\pi\eta D}. \quad (3)$$

b. Determination of density difference from sedimentation velocity

We apply Stokes' Law to a falling sphere in liquid to estimate the density difference between the spatially averaged protein aggregate and the buffer solution. For a sphere,

$$\Delta\rho = \frac{18\eta v}{g d^2}, \quad (4)$$

where $\Delta\rho$ is the difference in density between the particle and buffer, η is the viscosity, g is the gravitation constant 9.8 m/s^2, d is obtained from Equation (3), and the velocity v was obtained from a linear fit of the vertical position as a function of time.

Deviations from Stokes' Law due to wall effects and non-sphericity can be accounted for by generalizing Equations 3 and 4 cases to:

$$d' = \frac{kT}{3\pi\eta D\chi\lambda}, \quad (5)$$

$$\Delta\rho' = \frac{18\eta v\chi\lambda}{g d^2}, \quad (6)$$

where d (in Equation 6) represents the diameter of an equivalent volume sphere, λ is the wall factor [13,26] approximated by a model in which particles are falling midway between two semi-infinite walls separated by a distance L:

$$\lambda = \frac{1}{1-d/L} \quad (7)$$

and χ is the dynamic shape factor[27] which can be approximated as,

$$\chi = \frac{d_n+2d_s}{3d}, \quad (8)$$

with d_n and d_s the equivalent circular diameter of the particle's projection normal to its motion and the equivalent diameter of a sphere with the same surface area as the particle, respectively. The wall factor λ was included in our calculations, and found to have a 10 percent effect for particles with $d \approx 5$ μm, with reduced effect as d decreases. The value of χ was estimated for a variety of non-spherical shapes. For prolate/oblate spheroids with a

0.5 aspect ratio, $\chi = 0.96$ if the long axis is parallel to the flow, and $\chi = 1.07$ if the long axis is perpendicular to the flow. Particles were observed to tumble during sedimentation, so χ is effectively averaged over the path. Because the vast majority of particles had an aspect ratio >0.5, we did not include corrections for shape in the analysis.

RESULTS & DISCUSSION

Figure 2 shows examples of tracks obtained for the 1 μm to 5 μm diameter microspheres (Figure 2 a-d) and for the protein aggregates (Figure 2 e). For the 1 μm diameter microspheres in Figure 2 a, a selection of tracks longer than 4000 s is shown. These tracks exhibit substantial diffusive motion in combination with a net vertical displacement resulting from sedimentation. As the microsphere size increases, the net diffusive motion is reduced compared to the net vertical motion. For the 5 μm diameter microspheres in Figure 2 d—where tracks longer than 300 s are shown—the motion is primarily settling with a small amount of diffusive motion still evident in the track. This trend is due to the increased mass of the larger particle, which gives rise to a d^2 dependence of the settling velocity according to Stokes' law. Thus, for the smallest microsphere size, the challenge was to obtain good statistics for falling velocity measurement, while for the largest (5 μm) diameter microspheres, the challenge was to obtain good statistics for Brownian motion analysis for size measurement. Meeting these issues at the size extremes explains the variation in the number of tracks we analyzed at different bead sizes as shown in Table 1.

The protein aggregate tracks in Figure 2 e—where tracks longer than 200 s are shown—exhibit a variety of trajectories. Some of these trajectories resemble the 1 μm diameter microsphere tracks with substantial diffusion, while others resemble the 5 μm diameter microspheres dominated by settling motion.

Following MSD analysis of the tracks, key results were tabulated, including MSD diameter d, average equivalent spherical diameter from the image d_I, settling velocity v, density difference $\Delta\rho$ (a function of d and v), and density difference $\Delta\rho_I$ (a function of d_I and v). Tracks were sorted according to path length and histograms of the parameters for different path lengths were plotted. Some examples are shown in Figures 3 and 4.

For the 1 μm diameter microspheres, the MSD diameter distribution (Figure 3a) narrows with increasing path length. The microspheres do not become more monodisperse with increasing path length, but rather as there is more data in the longer paths, the precision of the measurement is improved. The median of the distributions is approximately 1 μm. In contrast, the medians for the distributions of the imaged diameters (Figure 3 b) are around 2.9 μm and the distribution widths have little dependence on path length. The discrepancy between the imaged diameter and known microsphere diameter is due to the aperture on the 20× lens used to increase the depth of field at the expense of increasing out-of-focus and diffraction effects. These effects affect the variable threshold algorithm. In flow imaging, it is typical that raw images of calibration microspheres will result in oversizing, and corrections can be made for this effect[28]. The velocity distribution (Figure 3 c) narrows for longer track lengths. Note that some particles exhibit negative velocity, especially for shorter track lengths. For these particles, the Brownian diffusion in the y-direction exceeds

the net settling displacement. Because the Brownian displacement is proportional to $t^{1/2}$ and the settling displacement is proportional to t—where t refers to the track length in seconds in this context—the settling displacement will eventually overtake the random Brownian displacement. Nevertheless, the medians of these distributions are meaningful and yield the average settling velocity for the 1 μm diameter microspheres.

For the 5 μm diameter microspheres, the increased mass and settling velocity reduces the time that a particular microsphere is in the field of view. In addition, the Brownian motion component is smaller in each time step, so there is less information to calculate the MSD diameter. This reduced information results in a larger MSD diameter spread, compared to the 1 μm diameter microspheres (Figure 4 a) and set a diameter value of 5 μm as an upper limit for MSD analysis of size in this experimental arrangement. We note that NTA and dynamic light scattering (DLS) face similar difficulties at larger sizes. Brownian motion has little effect on the net vertical displacement, resulting in a distribution that is determined by the settling velocity (Figure 4 c). The more rapid descent of particles through the field of view limits the track lengths to less than 500 s.

Based on the distributions obtained for all microsphere sizes, we used the following criteria for including tracks in the calculation of median values for important parameters:

$d \leq 4$ μm; Track length > 300 s
$d > 4$ μm; Track length > 200 s.
Results did not depend strongly on these criteria. For example, for nominal 2 μm diameter microspheres, with track length minima set to 50, 200, 300, 500, and 1000, we obtain median MSD diameters of 2.07 μm, 1.99 μm, 1.97 μm, 1.95 μm, and 1.94 μm; and median densities difference values of 0.0433 g/cm^3, 0.0468 g/cm^3, 0.0472 g/cm^3, 0.0474 g/cm^3, and 0.0468 g/cm^3, respectively.

The range of concentrations of the microspheres was explored to determine any effect of concentration (i.e. particle-particle interactions) on the results. The number of particles per frame ranged from 10 to 160. At the higher concentrations, there were a higher fraction of tracks with shorter lengths. This is due to the higher probability of tracks crossing, which, as mentioned above, ended the track for the purpose of analysis. Other than that, there was no effect on key results (MSD diameter, median image diameter, average density difference, etc.) for the range of concentrations measured.

Figure 5 shows the histograms of MSD diameters for protein aggregate data sorted by track length. This data represents tracks from 17 separate runs. As size is reduced below ≈1.5 μm, the number of particles in the distributions rapidly declines. This is a result of the measurement, not a reflection of the aggregate size distribution in the sample. Protein aggregates of this size are harder to detect by microphotography, as they approach the optical limits of detection and the small difference in the index of refraction with respect to the buffer make the particles even harder to detect[29]. Tracks of varying lengths were evaluated according to the above rule for microspheres. These tracks were then sorted by MSD size into 1 μm wide bins (i.e., from 0.5 μm to 1.5 μm, from 1.5 μm to 2.5 μm, and so on) with the aim of testing whether the average particle density is size-dependent or not.

As noted previously for the microspheres, the imaged size exceeds the calculated MSD size. Figure 6 shows results for the median imaged equivalent circular diameter (ECD) d_{Image} vs. the median MSD diameter d_{MSD} for both the microspheres and the protein aggregates, along with a dashed line showing $d_{\mathrm{Image}} = d_{MSD}$. Also in Figure 6 are quadratic fits for d_{MSD} as a function of d_{Image}:

$$d_{MSD} = a_0 + a_1 d_{\mathrm{Image}} + a_2 d_{\mathrm{Image}}^2 \qquad (9)$$

The fit parameters a_0, a_1, and a_2 for the two sets of data are shown in Figure 6. These fits could be used to correct an image diameter to the MSD diameter. In commercial instruments it is common to use a relation based on images collected from calibration microspheres in a properly tuned instrument, to yield a diameter that will obtain a correct value when imaging similar microspheres under the same conditions. The different fits in Figure 6 for aggregates compared to microspheres, show that fit parameters obtained from calibration beads would not yield optimal values when applied to images of protein aggregates. This is not surprising given the differences between the index of refraction for protein aggregates and polystyrene microspheres[28,29]. Note that the fit parameters in Figure 6 would only apply to the optical setup used in this experiment, which includes an aperture on the 20× lens designed to increase the depth of field for optimal particle tracking. It may be challenging—but possible—to obtain tracks of sufficient information content without the 2.5 mm aperture and with a 10× lens to obtain a more generalizable transformation function, but this was outside the scope of this work.

The median density difference $\Delta\rho$—calculated using both the MSD diameter and the imaged ECD—for the microspheres as a function of the MSD diameter is shown in Figure 7 along with a horizontal line representing the microsphere manufacturer's reported density of 1.050 g/cm^3 (which shows as a $\Delta\rho = 0.050$ g/cm^3). The results for the MSD-calculated density are approximately independent of microsphere size and in reasonable agreement with the manufacturer's given density, which serves as a reasonable validation of our experimental approach. As expected, the density calculated using the imaged ECD is substantially lower due to the significant oversizing effect evident from Figure 6.

The results for the dependence of aggregate average density on MSD-calculated size are shown in Figure 8 a, along with the results from polystyrene microspheres. As mentioned above, the data from Figure 5 was sorted into 1 μm wide bins, and the median density difference vs. the median MSD diameter is plotted for each bin. The error bars represent the standard error of the mean for each bin. The data shows the aggregate density difference decreases with increasing particle size, with a value of $\Delta\rho$=0.081 g/cm^3 at 1.37 μm and a value of $\Delta\rho$=0.029 g/cm^3 at 4.9 μm. Based on the measured density of the buffer solution (0.998817 g/cm^3), this gives a value for the average particle density of 1.080 g/cm^3 at 1.37 μm and a value of 1.028 g/cm^3 at 4.9 μm. If we assume the molecular density of the protein is 1.41 g/cm^3 (1410 kg/m^3) [6], our results suggest that the fraction of water in the stir-stressed NISTmAb aggregates ranges from 80 % at 1.37 μm to 93 % at 4.9 μm. These results indicate a lower density than has generally been assumed for subvisible particles,

and also suggests that when attempting to correlate resonance mass results with particle tracking or flow imaging—or when trying to estimate the amount of protein in aggregate form— a density in the range of 1.028 g/cm^3 to 1.080 g/cm^3 should be considered. Optimally, it may be necessary to consider the size-dependence of protein aggregate density. There may be differences in density based on the particular protein, stress method, and concentration of the protein monomer and excipients, which would be of interest for further study. We note that there may be some bias in the result for the lowest size bin (i.e., 0.5 μm to 1.5 μm). As mentioned in connection with Figure 5, we lose detection of the smaller particles more often (thus the median size for this bin is 1.37 μm) and, presumably, also particles with lower optical contrast (and therefore lower density). The result is that the density value 1.080 g/cm^3 at 1.37 μm may be seen as an upper bound. The density increase with decreasing size should not be surprising. Eventually, as the size regime moves down to the molecular scale, (dimers, trimers, etc.) the density will approach the pure protein density value (1.41 g/cm^3)[6]. In addition, we note that if the particle porosity allows liquid to flow through the particle during settling, this will enhance the sedimentation rate, thus causing the particle to appear to have a higher density. This is more likely for the larger particles, which would also suggest the value we obtain for 3 μm and above should be considered an upper bound.

Our results for $\Delta\rho$ can be compared with recent determinations of the average refractive index of aggregated proteins[28,30]. The difference in refractive index, Δn, of a particle relative to the matrix fluid is related to the difference in protein concentration, Δc, of the particle relative to the matrix fluid:

$$\Delta c \;=\; \Delta n \,/\, (\mathrm{d}n/\mathrm{d}c)\,, \tag{10}$$

where dn/dc refers to the refractive index increment and has a value of ≈ 0.189 cm^3/g [31]. The relationship between $\Delta\rho$ and Δc can be found using the fundamental definitions of these quantities. Suppose a quantity of protein of mass m_p and volume V_p are combined with a quantity of buffer of mass m_b and volume V_b to form a protein particle, and that the solution is ideal so that the combined particle volume is the sum $V_b + V_p$. Then,

$$\Delta\rho = \rho_p - \rho_b = \frac{m_p+m_b}{V_p+V_b} - \frac{m_b}{V_b}\,, \tag{11}$$

$$\Delta c = \frac{m_p}{V_p+V_b}\,, \tag{12}$$

where ρ_p and ρ_b are the densities of the dry protein (1.41 g/cm^3) and the buffer (1.0 g/cm^3), respectively. Solving for Δc,

$$\Delta c = \frac{\Delta\rho}{1-\rho_b/\rho_p}\,. \tag{13}$$

Combining Eqs. 10 and 13:

$$\Delta\rho = 1.54\ \Delta n\ \mathrm{g/cm^3}. \qquad (14)$$

Literature results for Δn values of protein aggregates formed by agitating polyclonal human IgG—in comparison to NISTmAb[28]—have been converted to $\Delta\rho$ and included in Figure 8 b. These values were obtained using quantitative phase imaging, which has poor accuracy for $d < 5$ μm due to the large correction factors and the difficulty in defining particle diameter. Even though the protein is different, the agreement with the more precise sedimentation results is quite good. Also shown are results based on a holographic determination of Δn[30]. These measurements—based on yet a third protein (bovine pancreas insulin)—demonstrate the same conclusions; that protein aggregates are highly hydrated and that $\Delta\rho$ climbs rapidly with decreasing particle diameter. Although our tracking method relies on photomicrographs, the determination of particle diameter and settling velocity does not rely on image analysis for particle dimensions, only for particle location. For that reason, the present results provide an orthogonal method to current image-analysis methods.

Observing that—even with monodisperse microspheres—the sedimentation/MSD analysis method used in this experiment results in apparent density distributions (as shown in Figure 3), we nevertheless sought to discern if there is a spread in densities among similarly-sized protein aggregates. To do this, we sampled protein aggregates of MSD-analyzed diameters from 1.8 μm to 2.2 μm. We did not observe enough size variation to significantly affect the density (as seen in Figure 8). We considered only tracks longer than 1000 steps for which error in the MSD size is reduced and the distribution has a better chance of reflecting something about the actual size distribution in the sample. This was also a diameter range where there were enough particles (177) to construct a density histogram. Figure 9 shows a comparison of this data with microspheres also sampled from measured diameters ranging from 1.8 μm to 2.2 μm (of which there were 624 particles). Note that we normalized the histograms to the total number of particles to allow for easy comparison. It is evident that the width of the protein aggregate density distribution is greater than that of the monodisperse polystyrene microspheres, suggesting that there is a real distribution of densities among aggregates of the same size.

CONCLUSIONS

In this work, we measured a spatially averaged density for protein aggregates by obtaining the sedimentation velocity and equivalent spherical hydrodynamic diameter based on mean square displacement analysis of particle tracks. Measurements using calibration microspheres suggest that the average results from a large number of particles correspond to the expected diameter and density of the microspheres. We find that tracking analysis is applicable up to sizes of 5 μm for an apparatus optimized for the micrometer size regime. MSD analysis of particle tracks—for which particle images are also acquired—may be used to correct for diffraction/out-of-focus effects in the particle images. The resulting relationships between imaged size and hydrodynamic size were shown to be different for polystyrene microspheres than for protein aggregates, likely due to differences in the index

of refraction. It may prove feasible to perform such correlation measurements using a flow microscopy instrument with conventional optics, rather than the altered configuration used in this work to optimize the density measurement. We found that the stir-stressed NISTmAb produced aggregates with average densities that decreased with hydrodynamic diameter: from 1.080 g/cm^3 at 1.37 μm to 1.028 g/cm^3 at 4.9 μm. These densities correspond to water fractions ranging from 80% to 93%. Finally, we found that there is a statistically significant broadening of the protein aggregate density distribution in the 1.8 μm to 2.2 μm range due to variation among the particles.

ACKNOWLEDGEMENTS

Thanks to Michael Carrier and John Schiel of the NIST Biomolecular Measurement Division for assistance with computer upgrades and for providing a sample of NIST reference material RM 8671, respectively. We would also like to acknowledge Sherry D. Scheckels of the NIST Physical Measurements Division for measurements of water and buffer density.

REFERENCES

1. Squire PG, Himmel ME 1979. Hydrodynamics and Protein Hydration. Arch Biochem Biophys 196(1):165-177.
2. Gekko K, Noguchi H 1979. Compressibility of Globular-Proteins in Water at 25-Degrees-C. J Phys Chem-Us 83(21):2706-2714.
3. Quillin ML, Matthews BW 2000. Accurate calculation of the density of proteins. Acta Crystallographica Section D 56(7):791-794.
4. Andersson KM, Hovmoller S 1998. The average atomic volume and density of proteins. Z Kristallogr 213(7-8):369-373.
5. Bernstein FC, Koetzle TF, Williams GJB, Meyer EF, Brice MD, Rodgers JR, Kennard O, Shimanouchi T, Tasumi M 1977. Protein Data Bank - Computer-Based Archival File for Macromolecular Structures. J Mol Biol 112(3):535-542.
6. Fischer H, Polikarpov I, Craievich AF 2004. Average protein density is a molecular-weight-dependent function. Protein Sci 13(10):2825-2828.
7. Fischer H, Neto MD, Napolitano HB, Polikarpov I, Craievich AF 2010. Determination of the molecular weight of proteins in solution from a single small-angle X-ray scattering measurement on a relative scale. J Appl Crystallogr 43:101-109.
8. Sung JJ, Pardeshi NN, Mulder AM, Mulligan SK, Quispe J, On K, Carragher B, Potter CS, Carpenter JF, Schneemann A 2015. Transmission Electron Microscopy as an Orthogonal Method to Characterize Protein Aggregates. J Pharm Sci-Us 104(2):750-759.
9. Wuchner K, Buchler J, Spycher R, Dalmonte P, Volkin DB 2010. Development of a Microflow Digital Imaging Assay to Characterize Protein Particulates During

Storage of a High Concentration IgG1 Monoclonal Antibody Formulation. J Pharm Sci-Us 99(8):3343-3361.
10. Barnard JG, Singh S, Randolph TW, Carpenter JF 2011. Subvisible Particle Counting Provides a Sensitive Method of Detecting and Quantifying Aggregation of Monoclonal Antibody Caused by Freeze-Thawing: Insights Into the Roles of Particles in the Protein Aggregation Pathway. J Pharm Sci-Us 100(2):492-503.
11. Kalonia C, Kumru OS, Prajapati I, Mathaes R, Engert J, Zhou SX, Middaugh CR, Volkin DB 2015. Calculating the Mass of Subvisible Protein Particles with Improved Accuracy Using Microflow Imaging Data. J Pharm Sci-Us 104(2):536-547.
12. Folzer E, Khan TA, Schmidt R, Finkler C, Huwyler J, Mahler HC, Koulov AV 2015. Determination of the Density of Protein Particles Using a Suspended Microchannel Resonator. J Pharm Sci-Us 104(12):4034-4040.
13. Bach LT, Riebesell U, Sett S, Febiri S, Rzepka P, Schulz KG 2012. An approach for particle sinking velocity measurements in the 3-400 mu m size range and considerations on the effect of temperature on sinking rates. Mar Biol 159(8):1853-1864.
14. Sediq AS, Waasdorp SKD, Nejadnik MR, van Beers MMC, Meulenaar J, Verrijk R, Jiskoot W 2017. Determination of the Porosity of PLGA Microparticles by Tracking Their Sedimentation Velocity Using a Flow Imaging Microscope (FlowCAM). Pharm Res-Dordr 34(5):1104-1114.
15. Zhao YL, Lai HSS, Zhang GL, Lee GB, Li WJ 2014. Rapid determination of cell mass and density using digitally controlled electric field in a microfluidic chip. Lab Chip 14(22):4426-4434.
16. Schiel JE, Davis DL, Borisov OV 2014. State-of-the-Art and Emerging Technologies for Therapeutic Monoclonal Antibody Characterization Volume 1. Monoclonal Antibody Therapeutics: Structure, Function, and Regulatory Space Preface. State-of-the-Art and Emerging Technologies for Therapeutic Monoclonal Antibody Characterization, Vol 1 - Monoclonal Antibody Therapeutics: Structure, Function, and Regulatory Space 1176:Ix-Xi.
17. Joubert MK, Luo QZ, Nashed-Samuel Y, Wypych J, Narhi LO 2011. Classification and Characterization of Therapeutic Antibody Aggregates. J Biol Chem 286(28):25118-25133.
18. International Association for the Properties of Water and Steam. IAPWS Formulation 2008 for the Viscosity of Ordinary Water Substance ed., http://www.iapws.org/relguide/visc.pdf.
19. Cavicchi RE, Collett C, Telikepalli S, Hu Z, Carrier M, Ripple DC 2017. Variable Threshold Method for Determining the Boundaries of Imaged Subvisible Particles. J Pharm Sci 106(6):1499-1507.
20. Schindelin J, Arganda-Carreras I, Frise E, Kaynig V, Longair M, Pietzsch T, Preibisch S, Rueden C, Saalfeld S, Schmid B, Tinevez JY, White DJ, Hartenstein V, Eliceiri K, Tomancak P, Cardona A 2012. Fiji: an open-source platform for biological-image analysis. Nat Methods 9(7):676-682.
21. Collins TJ 2007. ImageJ for microscopy. Biotechniques 43(1):25-+.
22. Tinevez JY, Perry N, Schindelin J, Hoopes GM, Reynolds GD, Laplantine E, Bednarek SY, Shorte SL, Eliceiri KW 2017. TrackMate: An open and extensible platform for single-particle tracking. Methods 115:80-90.

23. Savin T, Doyle PS 2007. Statistical and sampling issues when using multiple particle tracking. Phys Rev E 76(2).
24. Michalet X 2010. Mean square displacement analysis of single-particle trajectories with localization error: Brownian motion in an isotropic medium. Phys Rev E 82(4).
25. Ernst D, Kohler J 2013. Measuring a diffusion coefficient by single-particle tracking: statistical analysis of experimental mean squared displacement curves. Phys Chem Chem Phys 15(3):845-849.
26. Brenner H 1962. Effect of Finite Boundaries on the Stokes Resistance of an Arbitrary Particle. J Fluid Mech 12(1):35-48.
27. Leith D 1987. Drag on Nonspherical Objects. Aerosol Sci Tech 6(2):153-161.
28. Ripple DC, Hu ZS 2016. Correcting the Relative Bias of Light Obscuration and Flow Imaging Particle Counters. Pharm Res-Dordr 33(3):653-672.
29. Zolls S, Gregoritza M, Tantipolphan R, Wiggenhorn M, Winter G, Friess W, Hawe A 2013. How subvisible particles become invisible-relevance of the refractive index for protein particle analysis. J Pharm Sci 102(5):1434-1446.
30. Wang C, Zhong X, Ruffner DB, Stutt A, Philips LA, Ward MD, Grier DG 2016. Holographic Characterization of Protein Aggregates. J Pharm Sci-Us 105(3):1074-1085.
31. Zhao HY, Brown PH, Schuckt P 2011. On the Distribution of Protein Refractive Index Increments. Biophys J 100(9):2309-2317.

TABLE CAPTIONS

Table 1. Particle type and number of tracks measured.

FIGURE CAPTIONS

Figure 1. (a) Schematic of measurement setup. (b) Sample protein aggregate images and boundaries defined by the variable threshold algorithm.

Figure 2. Sample particle trajectories of polystyrene microspheres (a-d) and protein aggregates (e): (a) 1 μm microspheres, track lengths > 4000 s, (b) 2 μm microspheres, track lengths > 2350 s, (c) 3 μm microspheres, track lengths > 700 s, (d) 5 μm microspheres, track lengths > 300 s, (e) protein aggregates, track lengths > 200 s.

Figure 3. Histograms from track measurements of 1 μm microspheres for different track length ranges. (a) MSD diameter, (b) average image equivalent circular diameter (ECD), and (c) settling velocity.

Figure 4. Histograms from track measurements of 5 μm microspheres for different track length ranges. (a) MSD diameter, (b) average image equivalent circular diameter (ECD), and (c) settling velocity.

Figure 5. Histograms of MSD diameter from track measurements of protein aggregates for different track length ranges.

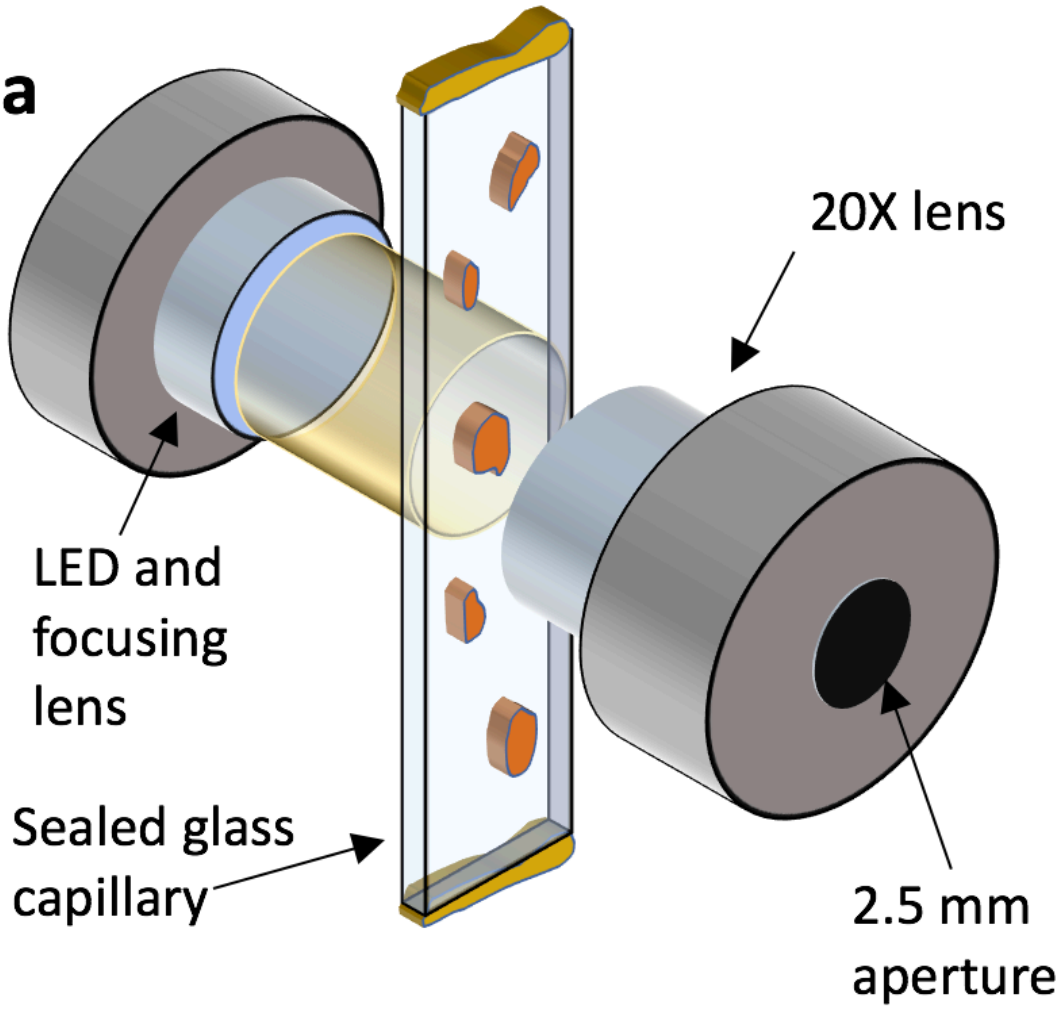
Figure 6. Image diameter vs. MSD diameter for polystyrene microspheres and protein aggregates. Error bars represent standard error of the mean with number of contributing points equal to 3335, 2839, 1268, 4437 for microspheres of nominal diameter 1 μm, 2 μm, 3 μm, and 5 μm, respectively. Listed are the coefficients for the fits to Equation 9 for the microsphere and protein aggregate data. The dashed line indicates the curve $d_{\mathrm{Image}} = d_{MSD}$.

Figure 7. Measured density difference between polystyrene microspheres and water based on the calculated MSD diameter (solid circles) and imaged equivalent circular diameter (plus symbols). The horizontal line indicates the microsphere manufacturer's quoted density of 1.050 g/cm^3.

Figure 8. (a) Measured density difference between protein aggregates and histidine buffer (solid squares) and polystyrene microspheres and water (solid circles) vs. MSD diameter. Error bars represent for the standard error of the mean for the microspheres listed in Figure 6. Errors for the protein aggregates also represent the standard error of the mean, with the number of contributing points equal to 255, 1644, 499, 121, and 55 for the aggregates in median diameter bins 1.37 μm, 2.0 μm, 2.8 μm, 3.8 μm, and 4.9 μm, respectively. (b) The density difference between protein aggregates and buffer for the NISTmAb aggregates measured by sedimentation (this work, squares), agitated IgG by quantitative phase imaging[28] (circles), and agitated bovine pancreas insulin by a holographic method [30] (triangles).

Figure 9. Normalized number of particles vs. density difference for protein aggregates and histidine buffer (solid squares) and polystyrene microspheres and water (solid circles).

Sample	Number of tracks analyzed
1 μm PSL	8486
2 μm PSL	5022
3 μm PSL	2584
5 μm PSL	6424
NISTmAb aggregate	9964


a
20X lens
LED and
focusing
lens
Sealed glass
capillary
2.5 mm
aperture

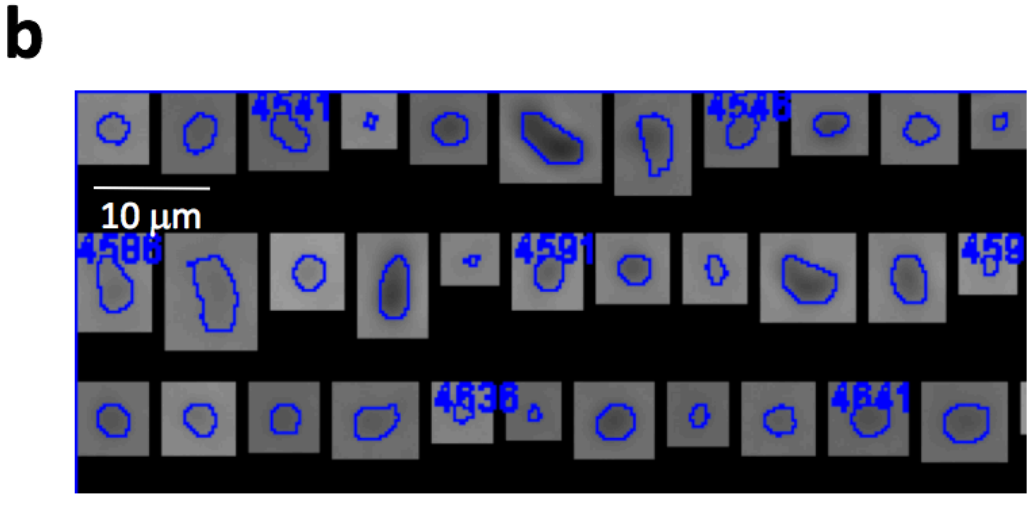

b
10 µm

Figure 1

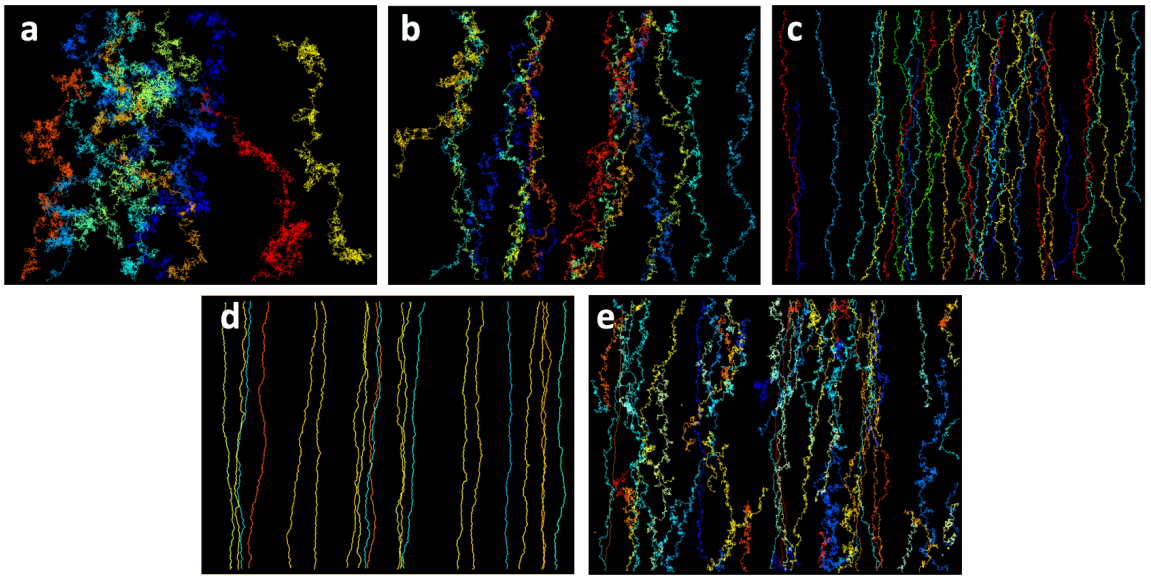


Figure 2

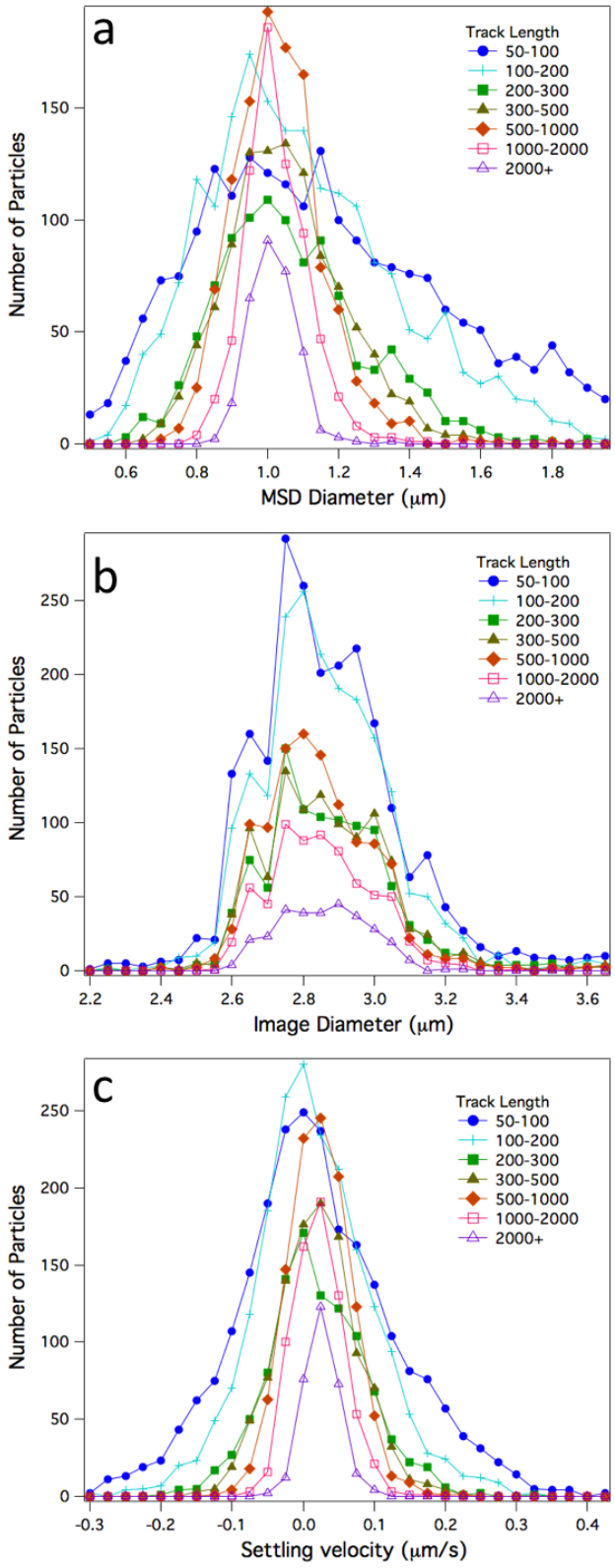

a
Track Length
50-100
100-200
200-300
300-500
500-1000
1000-2000
2000+
Number of Particles
MSD Diameter (μm)
b
Image Diameter (μm)
c
Settling velocity (μm/s)

Figure 3

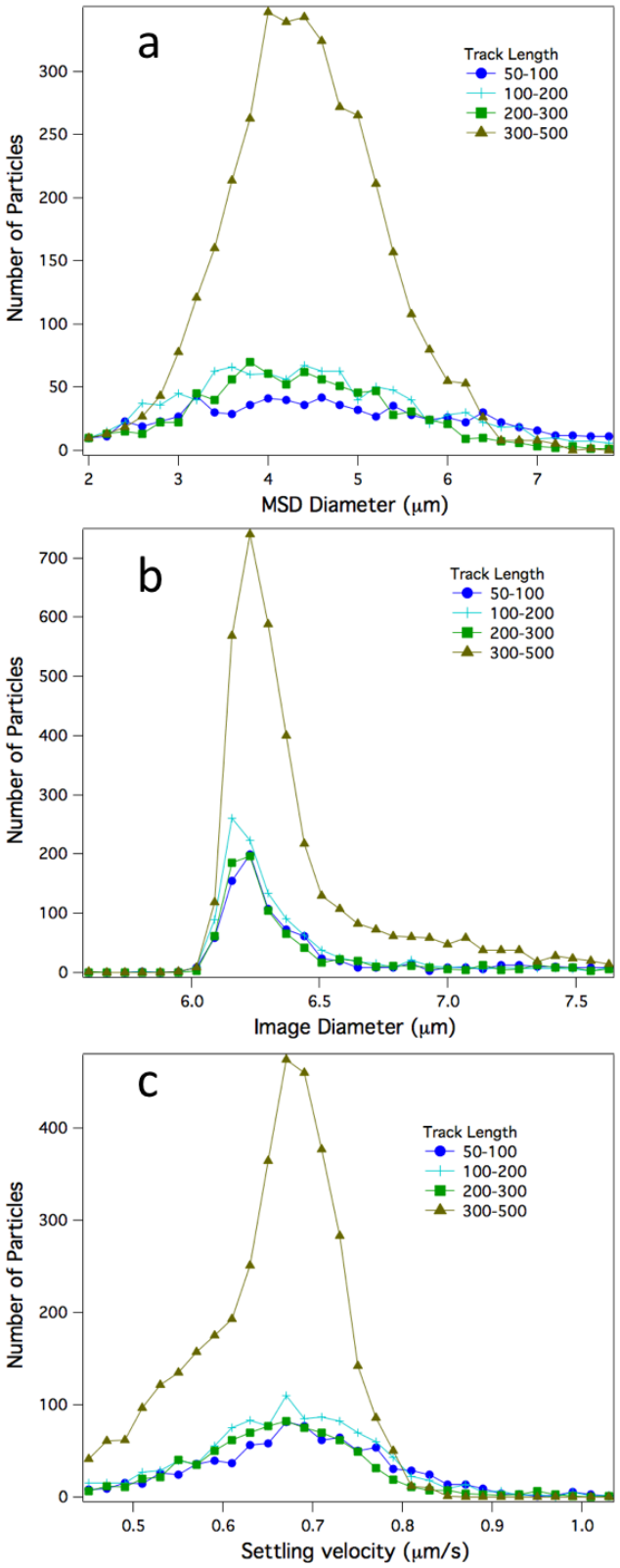

a
Track Length
50-100
100-200
200-300
300-500
Number of Particles
MSD Diameter (μm)
b
Track Length
50-100
100-200
200-300
300-500
Number of Particles
Image Diameter (μm)
c
Track Length
50-100
100-200
200-300
300-500
Number of Particles
Settling velocity (μm/s)

Figure 4

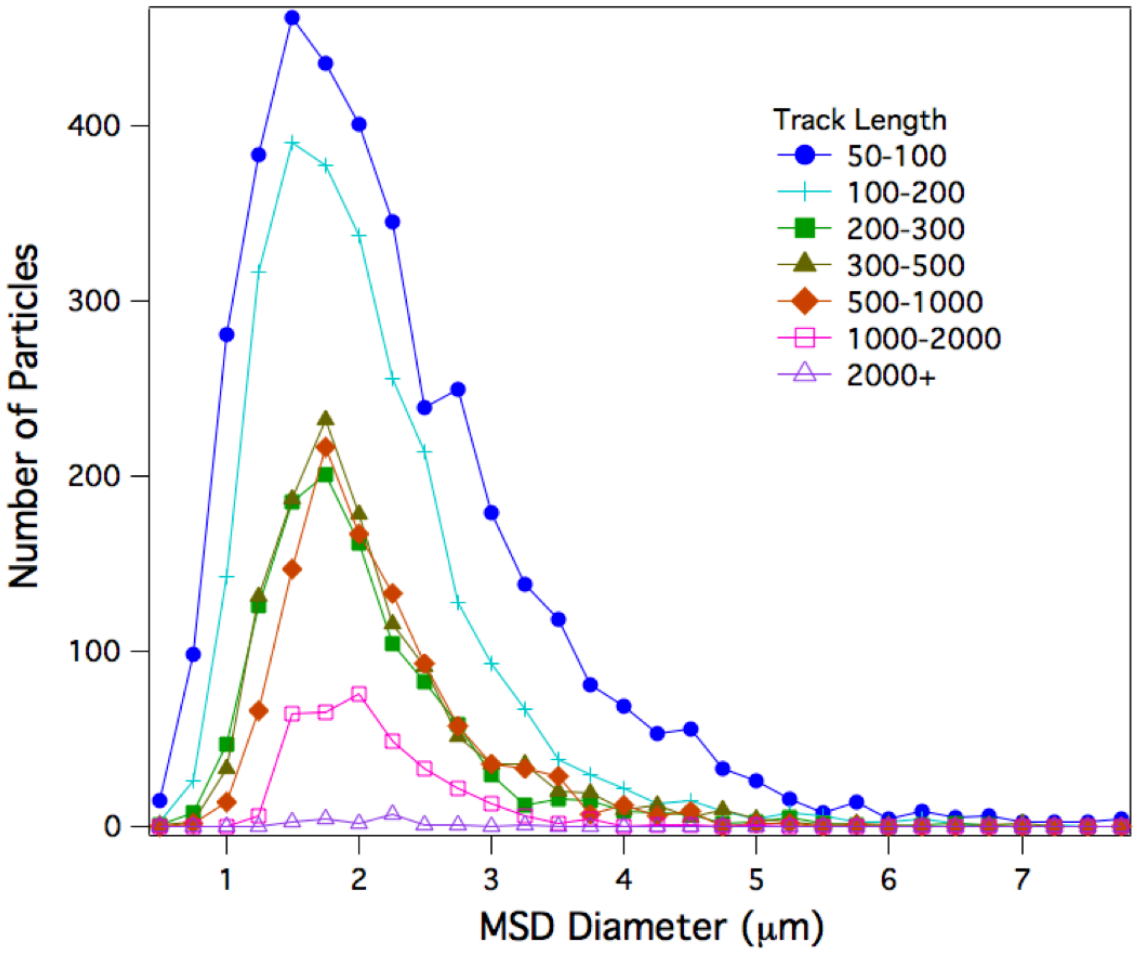

Track Length
50-100
100-200
200-300
300-500
500-1000
1000-2000
2000+
Number of Particles
0
100
200
300
400
1
2
3
4
5
6
7
MSD Diameter (μm)

Figure 5

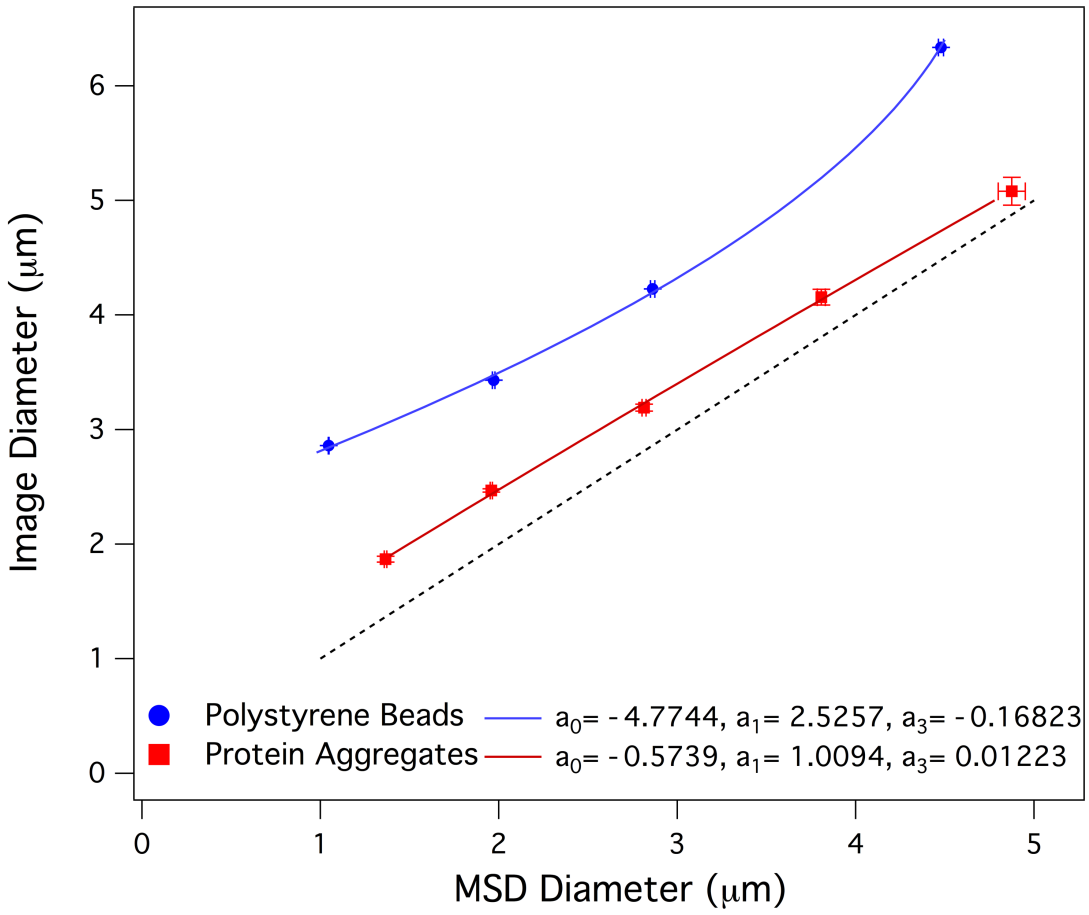


Image Diameter (μm)
MSD Diameter (μm)
0
1
2
3
4
5
6
Polystyrene Beads
Protein Aggregates
a0= -4.7744, a1= 2.5257, a3= -0.16823
a0= -0.5739, a1= 1.0094, a3= 0.01223

Figure 6

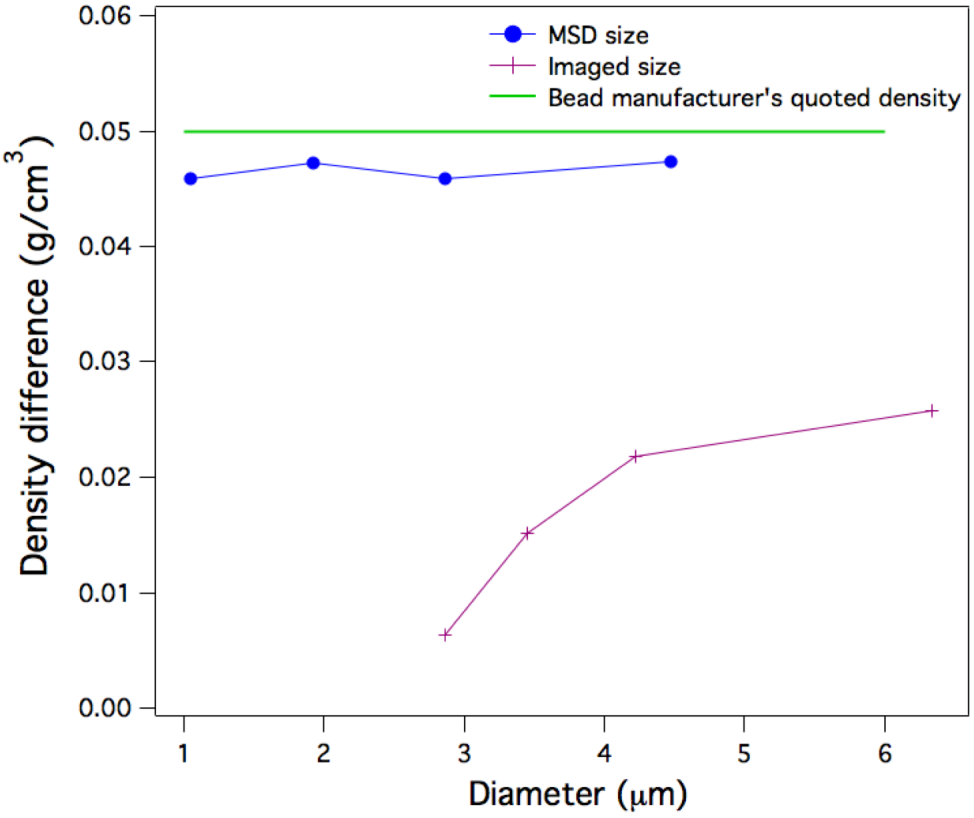

MSD size
Imaged size
Bead manufacturer's quoted density
Density difference (g/cm^3)
Diameter (μm)
0.06
0.05
0.04
0.03
0.02
0.01
0.00
1
2
3
4
5
6

Figure 7

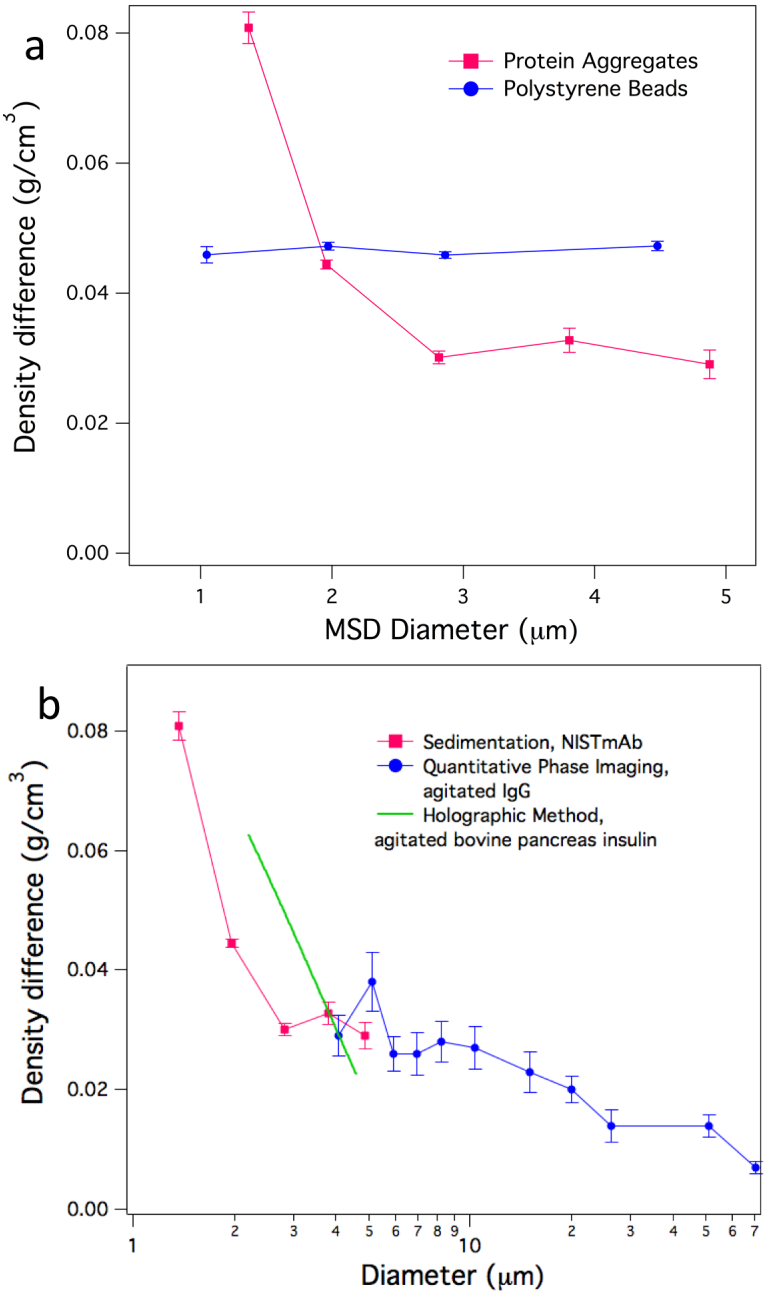

a
Protein Aggregates
Polystyrene Beads
Density difference (g/cm3)
0.08
0.06
0.04
0.02
0.00
1
2
3
4
5
MSD Diameter (μm)
b
Sedimentation, NISTmAb
Quantitative Phase Imaging, agitated IgG
Holographic Method, agitated bovine pancreas insulin
Density difference (g/cm3)
0.08
0.06
0.04
0.02
0.00
1
2
3
4
5
6
7
8
9
10
2
3
4
5
6
7
Diameter (μm)

Figure 8

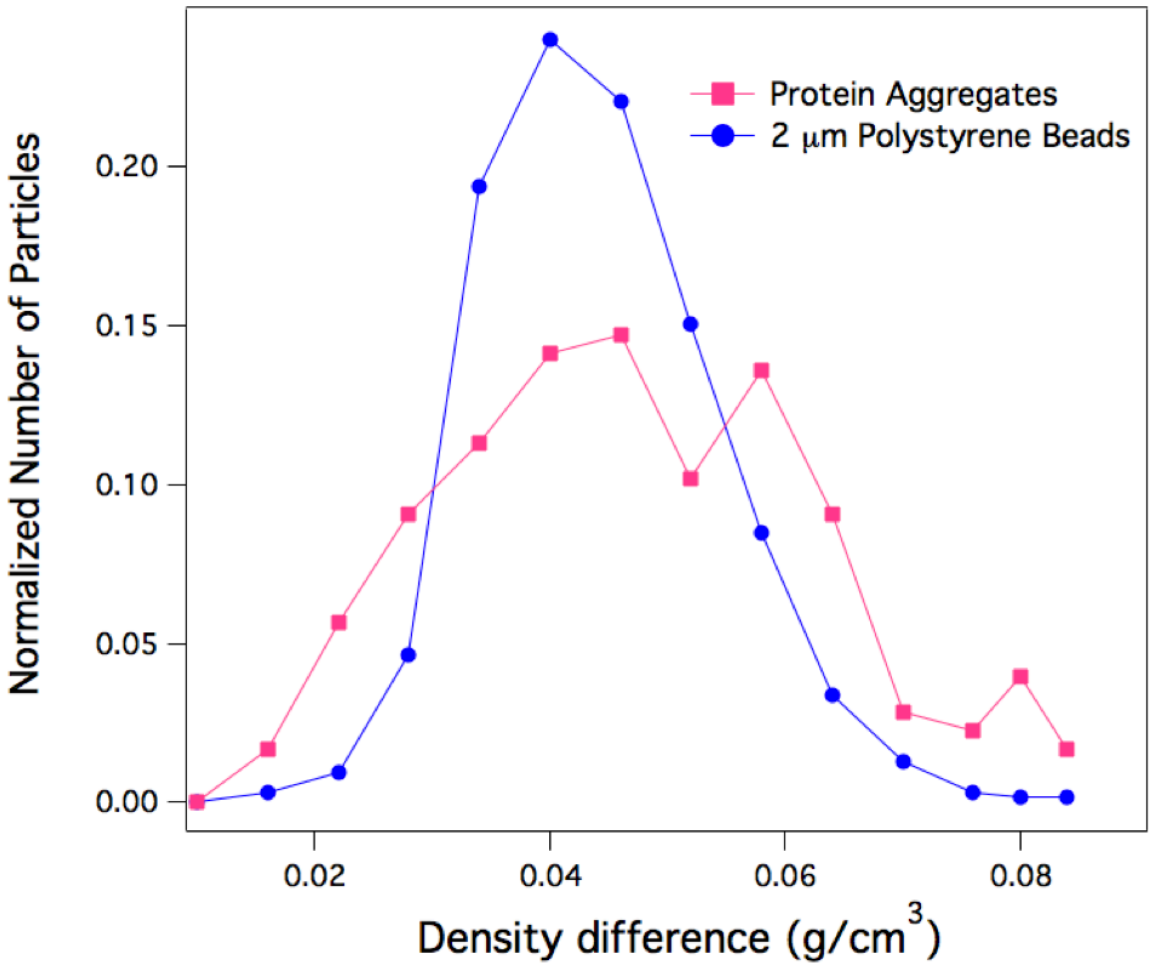

Protein Aggregates
2 μm Polystyrene Beads
Normalized Number of Particles
0.20
0.15
0.10
0.05
0.00
0.02
0.04
0.06
0.08
Density difference (g/cm3)

Figure 9

Supplementary:
This is a set of four computer programs used for data analysis in the paper "Measurement of Average Aggregate Density by Sedimentation" by Richard E. Cavicchi,
Jason King, and Dean C. Ripple
MakeXMLfile.py python program, converts a table of data into xml format for loading into Trackmate. Uses data file in csv format containing particle number, Area, Mean grayscale value of particle, XM, YM, and Slice for each particle analyzed by Variable Threshold or other particle analysis program.
ExportTracksToXML.java slightly modified version of Trackmate action file to allow for
export of particle grayscale value and radius along with x and y location
mainjb2.py python program for analyzing tracks to produce MSD diameter, density, and
a number of other particle properties from xml file produced by Trackmate
setup.py a set of functions used by mainjb2.py

```
# -*- coding: utf-8 -*-
"""PYTHON program "makexmlfile.py". Converts output of FIJI
ParticleAnalyzer into XML format usable by Trackmate
Columns are ParticleID Area Mean XM YM Slice, from Area, calculates
RADIUS, and also supplies Mean (brightness) as'QUALITY' in the output
XML file
"""
import numpy as np
import sys
import os
from xml.etree.ElementTree import ElementTree, XML
from xml.etree.ElementTree import Element
from xml.dom import minidom
import xml.etree.ElementTree as ET
def prettify(elem):
"""Return a pretty-printed XML string for the Element.
"""
rough_string = ET.tostring(elem, 'utf-8')
reparsed = minidom.parseString(rough_string)
return reparsed.toprettyxml(indent=" ")
fnameprefix = sys.argv[1]
fname = fnameprefix+'.csv'
ImageDataName=fnameprefix+'.tif'
print (os.getcwd())
ImageDataFolder=os.getcwd()
print (ImageDataFolder)
dtype1= np.dtype({'names': ['ParticleNumber','Area','Mean','XM',
'YM','Frame'],'formats': [int, int, int, float, float, int]})
a=np.loadtxt(fname,dtype=dtype1, delimiter=',',skiprows=1)
Spotstotal=len(a['Area'])
Spotsmaxindex=Spotstotal-1
framestotal=a[Spotsmaxindex]['Frame']
```

```
print ("number of spots",Spotstotal,"number of frames",framestotal)
root = Element('TrackMate')
tree=ElementTree(root)
model = Element('Model')
root.append(model)
root.set('version','2.8.1')
model.set('spatialunits','pixel')
model.set('timeunits','frame')
FeatureDeclarations=ET.SubElement(model,'FeatureDeclarations')
SpotFeatures=ET.SubElement(FeatureDeclarations,'SpotFeatures')
featurestr=['QUALITY','POSITION_X','POSITION_Y','POSITION_Z','POSITION
_T','FRAME','RADIUS','VISIBILITY']
namestr=['Quality','X','Y','Z','T','Frame','Radius','Visibility']
shortnamestr=['Quality','X','Y','Z','T','Frame','R','Visibility']
dimensionstr=['QUALITY','POSITION','POSITION','POSITION','TIME','NONE'
,'LENGTH','NONE']
isintstr=['false','false','false','false','false','true','false','true
']
Feature = [Element('Feature',feature=
featurestr[i],name=namestr[i],shortname=shortnamestr[i],dimension=dime
nsionstr[i],isint=isintstr[i])for i in range(8)]
SpotFeatures.extend(Feature)
EdgeFeatures=ET.SubElement(FeatureDeclarations,'EdgeFeatures')
TrackFeatures=ET.SubElement(FeatureDeclarations,'TrackFeatures')
AllSpots=ET.SubElement(model,'AllSpots')
AllSpots.set('nspots',str(Spotstotal))
s=0
for i in range(framestotal):
spotsinf=0
shold=s
while (a[s]['Frame']== i+1) and s<Spotsmaxindex:
spotsinf+=1
s=s+1
SpotsInFrame=ET.SubElement(AllSpots,'SpotsInFrame')
SpotsInFrame.set('frame',str(a[s-1]['Frame']-1))
Spot = [Element('Spot',ID=str(a[j]
['ParticleNumber']),VISIBILITY=str(1),POSITION_T=str(a[j]
['Frame']-1),POSITION_Z=str(0),POSITION_Y=str(a[j]
['YM']),RADIUS=str(np.sqrt(a[j]['Area']/np.pi)),FRAME=str(a[j]
['Frame']-1), POSITION_X=str(a[j]['XM']),QUALITY=str(a[j]['Mean']))
for j in range(shold,shold+spotsinf)]
SpotsInFrame.extend(Spot)
AllTracks=ET.SubElement(model,'AllTracks')
FilteredTracks=ET.SubElement(model,'FilteredTracks')
Settings = Element('Settings')
root.append(Settings)
ImageData=Element('ImageData',filename=ImageDataName,
folder=ImageDataFolder, width='1280', height='960', nslices='1',
nframes=str(framestotal), pixelwidth='1.0', pixelheight='1.0',
```

```
voxeldepth='1.0', timeinterval='1.0')
Settings.append(ImageData)
BasicSettings=Element('BasicSettings',xstart='0', xend='1279',
ystart='0', yend='959', zstart='0', zend='0', tstart='0',
tend=str(framestotal-1))
Settings.append(BasicSettings)
DetectorSettings=Element('DetectorSettings',DETECTOR_NAME='LOG_DETECTO
R', TARGET_CHANNEL='1', RADIUS='9.0', THRESHOLD='4.0',
DO_MEDIAN_FILTERING='false', DO_SUBPIXEL_LOCALIZATION='true')
Settings.append(DetectorSettings)
InitialSpotFilter=Element('InitialSpotFilter',feature='QUALITY',
value='0.0', isabove='true')
Settings.append(InitialSpotFilter)
SpotFilterCollection=ET.SubElement(Settings,'SpotFilterCollection')
TrackerSettings=ET.SubElement(Settings,'TrackerSettings')
TrackFilterCollection=ET.SubElement(Settings,'TrackFilterCollection')
AnalyzerCollection=ET.SubElement(Settings,'AnalyzerCollection')
SpotAnalyzers=ET.SubElement(AnalyzerCollection,'SpotAnalyzers')
SpotAnstr=['MANUAL_SPOT_COLOR_ANALYZER','Spot descriptive
statistics','Spot radius estimator','Spot contrast and SNR']
Analyzer = [Element('Analyzer',key= SpotAnstr[i])for i in range(4)]
SpotAnalyzers.extend(Analyzer)
EdgeAnalyzers=ET.SubElement(AnalyzerCollection,'EdgeAnalyzers')
EdgeAnstr=['Edge target','Edge mean location','Edge
velocity','MANUAL_EDGE_COLOR_ANALYZER']
Analyzer = [Element('Analyzer',key= EdgeAnstr[i])for i in range(4)]
EdgeAnalyzers.extend(Analyzer)
TrackAnalyzers=ET.SubElement(AnalyzerCollection,'TrackAnalyzers')
TrackAnstr=['Branching analyzer','Track duration','Track index','Track
location','Velocity','TRACK_SPOT_QUALITY']
Analyzer = [Element('Analyzer',key= TrackAnstr[i])for i in range(6)]
TrackAnalyzers.extend(Analyzer)
GUIState = Element('GUIState')
GUIState.set('state','ChooseTracker')
View=ET.SubElement(GUIState,'View')
View.set('key','HYPERSTACKDISPLAYER')
root.append(GUIState)
yy= prettify(root)
#print yy
fname = fnameprefix +'.xml'
ftest = open(fname,"w")
ftest.write(yy)
ftest.close()
#f.close()
package fiji.plugin.trackmate.action;
import fiji.plugin.trackmate.Logger;
import fiji.plugin.trackmate.Model;
import fiji.plugin.trackmate.Settings;
import fiji.plugin.trackmate.Spot;
```

```
import fiji.plugin.trackmate.TrackMate;
import fiji.plugin.trackmate.gui.TrackMateGUIController;
import fiji.plugin.trackmate.gui.TrackMateWizard;
import fiji.plugin.trackmate.io.IOUtils;
import fiji.plugin.trackmate.util.TMUtils;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.util.Set;
import java.util.TreeSet;
import javax.swing.ImageIcon;
import org.jdom2.Document;
import org.jdom2.Element;
import org.jdom2.output.Format;
import org.jdom2.output.XMLOutputter;
import org.scijava.plugin.Plugin;
/*
* PYTHON program. Modified version of ExportTracksToXML that
allows the export of particle mean brightness (as QUALITY) and
particle *particle radius as RADIUS.
These values are supplied to trackmate as a file in "LOAD a Trackmate
File"
*/
public class ExportTracksToXML extends AbstractTMAction {
public static final ImageIcon ICON = new
ImageIcon(TrackMateWizard.class.getResource("images/page_save.png"));
public static final String NAME = "Export tracks to XML file";
public static final String KEY =
"EXPORT_TRACKS_TO_XML_SIMPLE";
public static final String INFO_TEXT = "<html>" +
"Export the tracks in the current
model content to a XML " +
"file in a simple format. " +
"<p> " +
"The file will have one element per
track, and each track " +
"contains several spot elements.
These spots are " +
"sorted by frame number, and have 4
numerical attributes: " +
"the frame number this spot is in,
and its X, Y, Z position in " +
"physical units as specified in the
image properties. " +
"<p>" +
"As such, this format <u>cannot</u>
handle track merging and " +
"splitting properly, and is suited
```

```
only for non-branching tracks." +
"</html>";
private final TrackMateGUIController controller;
/*
* CONSTRUCTOR
*/
public ExportTracksToXML( final TrackMateGUIController
controller )
{
this.controller = controller;
}
/*
* METHODS
*/
/**
* Static utility that silently exports tracks in a simplified
xml format,
* describe in this class.
*
* @param model
* the {@link Model} that contains the tracks to
export.
* @param settings
* a {@link Settings} object, only used to read its
* {@link Settings#dt} field, the frame interval.
* @param file
* the file to save to.
* @throws FileNotFoundException
* if the target file cannot be written.
* @throws IOException
* if there is a problem writing the file.
*/
public static void export(final Model model, final Settings
settings, final File file) throws FileNotFoundException, IOException {
final Element root = marshall(model, settings,
Logger.VOID_LOGGER);
final Document document = new Document(root);
final XMLOutputter outputter = new
XMLOutputter(Format.getPrettyFormat());
outputter.output(document, new
FileOutputStream(file));
}
@Override
public void execute(final TrackMate trackmate) {
logger.log("Exporting tracks to simple XML format.
\n");
final Model model = trackmate.getModel();
final int ntracks =
model.getTrackModel().nTracks(true);
```

```
if (ntracks == 0) {
logger.log("No visible track found. Aborting.
\n");
return;
}
logger.log(" Preparing XML data.\n");
final Element root = marshall(model,
trackmate.getSettings(), logger);
File folder;
try {
folder = new
File(trackmate.getSettings().imp.getOriginalFileInfo().directory);
} catch (final NullPointerException npe) {
folder = new
File(System.getProperty("user.dir")).getParentFile().getParentFile();
}
File file;
try {
String filename =
trackmate.getSettings().imageFileName;
final int dot = filename.indexOf(".");
filename = dot < 0 ? filename :
filename.substring(0, dot);
file = new File(folder.getPath() +
File.separator + filename +"_Tracks.xml");
} catch (final NullPointerException npe) {
file = new File(folder.getPath() +
File.separator + "Tracks.xml");
}
file = IOUtils.askForFileForSaving(file,
controller.getGUI(), logger);
if (null == file) {
return;
}
logger.log(" Writing to file.\n");
final Document document = new Document(root);
final XMLOutputter outputter = new
XMLOutputter(Format.getPrettyFormat());
try {
outputter.output(document, new
FileOutputStream(file));
} catch (final FileNotFoundException e) {
logger.error("Trouble writing to "+file+":\n"
+ e.getMessage());
} catch (final IOException e) {
logger.error("Trouble writing to "+file+":\n"
+ e.getMessage());
}
logger.log("Done.\n");
```

```
}
private static Element marshall(final Model model, final
Settings settings, final Logger logger) {
logger.setStatus("Marshalling...");
final Element content = new Element(CONTENT_KEY);
content.setAttribute(NTRACKS_ATT,
""+model.getTrackModel().nTracks(true));
content.setAttribute(PHYSUNIT_ATT,
model.getSpaceUnits());
content.setAttribute(FRAMEINTERVAL_ATT,
""+settings.dt);
content.setAttribute(FRAMEINTERVALUNIT_ATT,
""+model.getTimeUnits());
content.setAttribute(DATE_ATT,
TMUtils.getCurrentTimeString());
content.setAttribute(FROM_ATT,
TrackMate.PLUGIN_NAME_STR + " v" + TrackMate.PLUGIN_NAME_VERSION);
final Set<Integer> trackIDs =
model.getTrackModel().trackIDs(true);
int i = 0;
for (final Integer trackID : trackIDs) {
final Set<Spot> track =
model.getTrackModel().trackSpots(trackID);
final Element trackElement = new
Element(TRACK_KEY);
trackElement.setAttribute(NSPOTS_ATT,
""+track.size());
// Sort them by time
final TreeSet<Spot> sortedTrack = new
TreeSet<Spot>(Spot.timeComparator);
sortedTrack.addAll(track);
for (final Spot spot : sortedTrack) {
final int frame =
spot.getFeature(Spot.FRAME).intValue();
final double x =
spot.getFeature(Spot.POSITION_X);
final double y =
spot.getFeature(Spot.POSITION_Y);
final double z =
spot.getFeature(Spot.RADIUS);
final double ii =
spot.getFeature(Spot.QUALITY);
final Element spotElement = new
Element(SPOT_KEY);
spotElement.setAttribute(T_ATT,
""+frame);
spotElement.setAttribute(X_ATT,
""+x);
spotElement.setAttribute(Y_ATT,
```

```
""+y);
spotElement.setAttribute(Z_ATT,
""+z);
spotElement.setAttribute(I_ATT,
""+ii);
trackElement.addContent(spotElement);
}
content.addContent(trackElement);
logger.setProgress(i++ / (0d +
model.getTrackModel().nTracks(true)));
}
logger.setStatus("");
logger.setProgress(1);
return content;
}
/*
* XML KEYS
*/
private static final String CONTENT_KEY = "Tracks";
private static final String DATE_ATT =
"generationDateTime";
private static final String PHYSUNIT_ATT = "spaceUnits";
private static final String FRAMEINTERVAL_ATT =
"frameInterval";
private static final String FRAMEINTERVALUNIT_ATT =
"timeUnits";
private static final String FROM_ATT = "from";
private static final String NTRACKS_ATT =
"nTracks";
private static final String NSPOTS_ATT = "nSpots";
private static final String TRACK_KEY = "particle";
private static final String SPOT_KEY = "detection";
private static final String X_ATT = "x";
private static final String Y_ATT = "y";
private static final String Z_ATT = "z";
private static final String T_ATT = "t";
private static final String I_ATT = "i";
@Plugin( type = TrackMateActionFactory.class )
public static class Factory implements TrackMateActionFactory
{
@Override
public String getInfoText()
{
return INFO_TEXT;
}
@Override
public String getName()
{
return NAME;
```

```
}
@Override
public String getKey()
{
return KEY;
}
@Override
public TrackMateAction create( final
TrackMateGUIController controller )
{
return new ExportTracksToXML( controller );
}
@Override
public ImageIcon getIcon()
{
return ICON;
}
}
}
# -*- coding: utf-8 -*-
# mainjb2.py program for analyzing tracks supplied by xml file from
trackmate
# and tstamp file made (two columns, frame, timestamp)
import xml.etree.ElementTree as ET
import sys
import math
import csv
import pandas as pd
from scipy.optimize import curve_fit
from setup import stick_check
from setup import calc_dia
from setup import offset
from setup import StandardError
from setup import calc_density
from setup import np
def linear_func(t, D, b):
return D*t + b
def get_msd_point(df):
return pd.Series(data=[df['sd'].mean(), df.shape[0]],
index=['msd', 'count'])
def get_msd(x, y, t, min_delta=1, max_delta=None):
'''Return msd points binned by time intervals calculated from
timestamps.'''
result = []
if not max_delta:
max_delta = x.shape[0]
for delta in np.arange(min_delta, max_delta):
pos = 0
while pos + delta < int(x.shape[0]):
```

```
dx2 = np.power(x[pos+delta] - x[pos], 2)
dy2 = np.power(y[pos+delta] - y[pos], 2)
dr2 = dx2 + dy2
dt = t[pos+delta] - t[pos]
result.append([delta, dr2, dt])
pos += 1
result = np.array(result)
# Bin squared displacements by dt and calc msd points
df = pd.DataFrame(result, columns=['delta','sd','dt'])
df =
df.groupby(df['dt'].round(6)).apply(get_msd_point).reset_index()
return df
def process_track(x, y, t, min_delta=1, max_delta=10):
'''Return diffusion coefficient from particle track data.'''
df = get_msd(x, y, t, min_delta, max_delta)
msd = df['msd'].values
dt = df['dt'].values
sigma = (df['count'].max()/df['count']).values
# Fit line to msd points weighted by bin population
popt, pcov = curve_fit(linear_func, dt, msd, sigma=sigma)
return popt[0]/4
fname = sys.argv[1]
fnamets = fname[:(fname.find("_"))]+'TstampO.csv'
reader = csv.DictReader(open(fnamets))
dtype1= np.dtype({'names': ['Frame','Tstamp'],'formats': [int,
float]})
frmtim=np.loadtxt(fnamets,dtype=dtype1, delimiter=',',skiprows=0)
prefix=fname[:5]+'_'
#%%
## Set experimental parameters
nMin = 50
T = 273.15 + 26.4
eta = 0.0071137-2.0857e-5*T
scale = 0.28
dt = 1/.976
# time interval in seconds5
## These parameters should set to True if a drift velocity is present
## I recommend leaving y_drift as True for sedimentation experiments.
x_drift = False
y_drift = True
#%%
## Read datafile and scale x-, y-, and t- measurements
print ('Reading datafile and scaling measurements.')
tree = ET.parse(fname)
root = tree.getroot()
data = []
nTracks = np.int(root.attrib['nTracks'])
idx = 0
for j in range(nTracks):
```

```
Track = j+1
nSpots = np.int(root[j].attrib['nSpots'])
# print nSpots
for i in range(nSpots):
time = np.float(root[j][i].attrib['t'])
x = np.float(root[j][i].attrib['x'])
y = np.float(root[j][i].attrib['y'])
z = np.float(root[j][i].attrib['z'])
i = np.float(root[j][i].attrib['i'])
data.append([x, y, z, i,time, Track])
idx += 1
data = np.array(data)
##data[:,4] *= dt
data[:,0:2] *= scale
data[:,2] *= 2*scale
## Step through individual tracks and perform msd analysis
## The length parameter is the length number of points in the msd
curve used for fitting to determine diff const
Aparticle=[]
An_orig=[]
An=[]
Ax_0=[]
Ay_0=[]
At_0=[]
AdiaMIN=[]
AdiaMAX=[]
AdiaMED=[]
AdiaMaxInten=[]
AintMED=[]
AMaxIntenFrame=[]
Adia=[]
Adia_error=[]
Av_drift=[]
Av_drift_error=[]
Avx=[]
Adensity_MIN =[]
Adensity_MaxInten =[]
Adensity_MED =[]
Adensity_msd=[]
Adensity_msd_error=[]
## A length of 10 gives good results for 1 um - 10 um diameter
particles
length = 10
particle=1
particles = np.unique(data[:,-1])
f = open(fname.split('.')[0] + '_out.csv', 'w')
#f.write('Track #, Length, Final_Length, x, y, t,
dia_fc,diaMED,diaMaxInten,MaxIntenFrame, dia_msd,\
# dia_msd_error, v_drift, density_MaxInten , density_msd,
```

```
density_msd_error\n')
f.write(prefix+'Track #,'+ prefix+'Length,'+ prefix+'Final_Length,'+
prefix+'x,'+ prefix+'y,'+ prefix+'t,'+ prefix+'intMED,'+
prefix+'diaMIN,'+
prefix+'diaMAX,'+prefix+'diaMED,'+prefix+'diaMaxInten,'+prefix+'MaxInt
enFrame,'+ prefix+'dia_msd,'+prefix+'dia_msd_error,'+
prefix+'v_drift,'+ prefix+'v_drift_error,'+ prefix+'vx,'+
prefix+'density_diaMIN,'+ prefix+'density_diaMED,'+
prefix+'density_MaxInten,'+ prefix+'density_msd,'+
prefix+'density_msd_error\n')
for particle in particles:
print ('{:d}'.format(int(particle)))
## Find all positions for a given run.
idx = np.where(data[:,-1] == particle)[0]
dia_fc = data[idx, 2]
i_fc = data[idx, 3]
x = data[idx, 0]
y = data[idx, 1]
t = data[idx, 4]
tframe = t.astype(int)
lentfr=len(tframe)
t_0 = tframe[0]
for j,tfrm in enumerate(tframe):
i=0
while (frmtim[tfrm+i]['Frame'] != tfrm) :
i += 1
if ((i+tfrm)>lentfr-1):
break
t[j] = frmtim[tfrm+i]['Tstamp']
## print("j",j,"i",i,"tfrm",tfrm,"t",t[j])
## print(tframe[0],t[0])
## print(lentfr-1,tframe[lentfr-1],t[lentfr-1])
n_orig = x.size
## Determine if particle is stuck or sticks during the run.
## If it sticks, truncate the run to the point in time where it
sticks.
a = stick_check(x, y, length = 6, threshold = 0.28)
if a:
## Modify this programe, if particle is stuck truncate all so track
will be excluded
## x = x[:a[0]]
## y = y[:a[0]]
## t = t[:a[0]]
## i_fc=i_fc[:a[0]]
x = x[:1]
## Check the length of the truncated/untruncated run. Only perform
msd
## analysis on runs that equal or exceed nMin.
n = x.size
```

```
if n >= nMin:
x_0 = x[0]/scale
y_0 = y[0]/scale
## t_0 = int(t[0]/dt)
## This is specifically for settling experiments. Calculate
the
## average y-velocity over the entire run.
## v_drift = (y[-1] - y[0])/(t[-1] - t[0])
v_drift,b = np.polyfit(t, y, 1)
## Compensate for a steady drift velocity if flagged. This is
very
## important. If drift velocity is present and not corrected
for, msd
## analysis WILL fail.
if x_drift == True:
x = offset(x, t)
if y_drift == True:
y = offset(y, t)
diff_const = process_track(x, y, t, min_delta=1,
max_delta=length)
dia = calc_dia(diff_const, eta, T)
diff_length = math.sqrt(2*diff_const*(t[-1] - t[0]))
v_drift_error = diff_length/(t[-1] - t[0])
vx = (x[-1] - x[0])/(t[-1] - t[0])
SE = StandardError(diff_const, x, length-1)
dia_error = np.abs(calc_dia(diff_const + SE, eta, T) - dia)
##wall effect correction factor wallfact
wallfact = 1/(1-dia/50)
## Calculate density from both the flowcam-reported diameter
(the
## minimum value of all reported diameters for the run) and
the
## msd-derived diameter
density_msd = calc_density(dia, v_drift, eta)
density_msd *= wallfact
dia /= wallfact
density_msd_error = np.abs(calc_density(dia + dia_error,
v_drift+v_drift_error, eta) - density_msd)
MaxIntenFrame= np.argmax(i_fc)
diaMaxInten=dia_fc[MaxIntenFrame]
diaMED=np.median(dia_fc)
diaMIN = np.min(dia_fc)
diaMAX = np.max(dia_fc)
intMED = np.median(i_fc)
density_MaxInten = calc_density(diaMaxInten, v_drift, eta)
density_Med = calc_density(diaMED, v_drift, eta)
density_Min = calc_density(diaMIN, v_drift, eta)
Aparticle.append(particle)
An_orig.append(n_orig)
```

```
An.append(n)
Ax_0.append(x_0)
Ay_0.append(y_0)
At_0.append(t_0)
AintMED.append(intMED)
AdiaMIN.append(diaMIN)
AdiaMAX.append(diaMAX)
AdiaMED.append(diaMED)
AdiaMaxInten.append(diaMaxInten)
AMaxIntenFrame.append(MaxIntenFrame)
Adia.append(dia)
Adia_error.append(dia_error)
Av_drift.append(v_drift)
Av_drift_error.append(v_drift_error)
Avx.append(vx)
Adensity_MaxInten.append(density_MaxInten )
Adensity_MIN.append(density_Min )
Adensity_MED.append(density_Med )
Adensity_msd.append(density_msd)
Adensity_msd_error.append(density_msd_error)
# f.write('%d, %d, %d, %0.3f, %0.3f, %d, %0.3f,%0.3f, %0.3f,
%d, %0.3f, %0.3f, %0.3f,\
# %0.3f, %0.3f, %0.3f\n' % (particle, n_orig, n, x_0,
y_0,\
# t_0, diaMIN,diaMED, diaMaxInten,MaxIntenFrame,dia,
dia_error, v_drift, density_MaxInten ,\
# density_msd, density_msd_error))
# print (diff_const)
else:
print ('Track too short.')
q=len(At_0)
for j in range(q):
idx = np.argsort(Adia)
Aparticle = np.array(Aparticle)[idx]
An_orig= np.array(An_orig)[idx]
An= np.array(An)[idx]
Ax_0= np.array(Ax_0)[idx]
Ay_0= np.array(Ay_0)[idx]
At_0= np.array(At_0)[idx]
AdiaMIN= np.array(AdiaMIN)[idx]
AdiaMAX= np.array(AdiaMAX)[idx]
AdiaMED= np.array(AdiaMED)[idx]
AdiaMaxInten= np.array(AdiaMaxInten)[idx]
AMaxIntenFrame= np.array(AMaxIntenFrame)[idx]
Adia= np.array(Adia)[idx]
Adia_error= np.array(Adia_error)[idx]
Av_drift= np.array(Av_drift)[idx]
Av_drift_error= np.array(Av_drift_error)[idx]
Avx= np.array(Avx)[idx]
```

```
Adensity_MIN = np.array(Adensity_MIN)[idx]
Adensity_MED = np.array(Adensity_MED)[idx]
Adensity_MaxInten = np.array(Adensity_MaxInten)[idx]
Adensity_msd= np.array(Adensity_msd)[idx]
Adensity_msd_error= np.array(Adensity_msd_error)[idx]
f.write ('%d, %d, %d, %0.3f,%0.3f, %d, %d, %0.3f,%0.3f,%0.3f,
%0.3f, %d,%0.3f, %0.3f, %0.3f, %0.3f,%0.3f,%0.3f, %0.3f, %0.3f, %0.3f,
%0.3f\n'\
% (Aparticle[j], An_orig[j], An[j],
Ax_0[j],Ay_0[j],At_0[j],AintMED[j], AdiaMIN[j], AdiaMAX[j],AdiaMED[j],
AdiaMaxInten[j],\
AMaxIntenFrame[j],Adia[j],Adia_error[j], Av_drift[j],
Av_drift_error[j],Avx[j], Adensity_MIN [j],Adensity_MED
[j],Adensity_MaxInten [j],Adensity_msd[j], Adensity_msd_error[j]))
f.close()
# -*- coding: utf-8 -*-
#This is setup.py a set of python functions used by the main program
for analyzing tracks
import numpy as np
import scipy.stats
from scipy import constants
def split_array(x, length, stagger):
"""
split array x in chunks of length with stagger (amount of overlap)
"""
L = [x[i:i+length] for i in range(0,len(x),int(stagger))]
L = [xx for xx in L if len(xx)==length]
return np.array(L)
def stick_check(x, y, length = 6, threshold = 0.1):
xs = split_array(x, length, 1)
ys = split_array(y, length, 1)
idx = []
for i in range(len(xs)):
dx = xs[i] - xs[i][0]
dy = ys[i] - ys[i][0]
if np.all(np.abs(dx) < threshold) and np.all(np.abs(dy) <
threshold):
idx.append(i)
return idx
def calc_dia(D, eta = 0.00089, T = 298.15):
'''
Calculate hydrodynamic diameter from Stokes-Einstein equation.
Needs diffusion coefficient, dynamic viscosity [N*s/m^2], and
temperature [K].
'''
D *= 1E-12 # m^2/s
r = constants.k*T/(6*constants.pi*eta*D)
result = 2*r*1E6 # microns
return result
```

```
def offset(y, t):
v_avg = (y[-1] - y[0])/(t[-1] - t[0])
result = y - v_avg*t
return result
def StandardError(D, x, n):#, conf=0.95, ddof=1):
"""
Calculate the standard error over track.
"""
SE = D*np.sqrt((2*n)/(3*(x.shape[0]-n)))
return SE
def calc_density(d, v_drift, eta = 0.00089):
'''
Returns delta rho.
Needs a hydrodynamic diameter [µm], average velocity [µm/s], and
dynamic viscosity [N*s/m^2].
'''
r = 0.5*d
r *= 1E-6 # m
v_drift *= 1E-6 # m/s
result = (9*eta*v_drift)/(2*constants.g*r**2)
return result
```