

Pythonによる画像解析を用いたタンパク質濃縮相内の線維核形成検出

小林 恒一*, 山内 皓太*, 福山 真央

(* : equal contribution)

1 はじめに

近年、目覚ましいデータサイエンスの発達により、分析化学分野においてもDXを利用した先駆的な論文が多く報告されている^{1)~3)}。自分の研究とプログラミングを組み合わせて考えることを考えてみたことのある読者は多いのではないかと思う。

最近プログラミングに関する専門的な教育を受けずとも、画像解析や機械学習といった複雑なデータ解析のスク립トを簡単に書くことができるようになってきた。特にPython⁴⁾の出現により初心者にとってのプログラミングのハードルが明らかに下がったと著者らは感じる。Pythonには下記のような長所がある。

- ①他のプログラミング言語と比べて構文が非常にシンプルで直感的であり、スク립トの中に意図のわからない記述がほとんどない。また、文法の柔軟性も高いため、非常に使いやすい言語である。
- ②多様なライブラリが用意されている。機械学習や画像解析など専門的なもの以外にも、csvファイル操作や、多次元データの計算など、細やかなところまでライブラリが用意されている。

そのため、Pythonの基本的な文法を理解して、上記ライブラリに用意された関数のインプットとアウトプットを正確に把握すれば、プログラミング初心者でも様々なデータ解析が可能である。実際筆者らもほとんどプログラミング教育を受けたことがなかったが(大学の学部時代に1, 2コマ受講した程度)、最近Pythonを用いて画像解析を行い、論文を執筆した⁵⁾。

本稿では、最近筆者らが報告した画像解析によるアミロイド核生成の定量に関する研究⁵⁾を例として取り上げ、Pythonを用いた共焦点蛍光顕微画像の解析について説明する。本稿では主に画像解析が必要となるライブラリや関数のコンセプトを説明する。Pythonの基本的な文法についてはウェブサイト等⁶⁾を参照されたい。

2 アミロイド核生成定量に関する研究の概要

2.1 研究背景

アミロイド線維とはタンパク質が1次元方向に規則正しく集積した凝集体であり、アルツハイマー病などの神経変性疾患に関連している⁷⁾⁸⁾。しかし、アミロイド

線維の生成過程の機構については未解明な部分が多いのが現状である。アミロイド線維の形成は、タンパク質の結晶化と同じように nucleation-dependent モデルで説明される⁹⁾¹⁰⁾。このモデルでは、核形成は非常に低頻度であるが、一旦核形成が起こると線維の伸長は速やかに進行する。

近年の研究から多くのアミロイド生成タンパク質が細胞内で液-液相分離 (liquid-liquid phase separation, LLPS) を起こし、高濃度な液状集合体(ここでは濃縮相とよぶ)を形成することが分かってきた¹¹⁾。LLPSによって形成される濃縮相中のタンパク質濃度は極めて高いため、濃縮相中でアミロイド核が生成するのではないかと考えられ始めた¹²⁾(図1)。

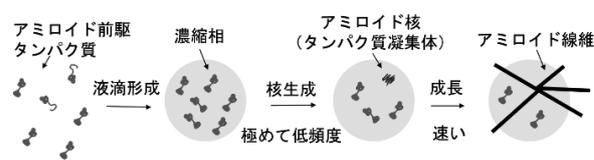


図1 濃縮相形成を介したアミロイド線維伸長過程

そこで筆者らは、濃縮相中のアミロイド核生成をイメージングにより定量的に解析する方法を提案した⁵⁾。マイクロメートルサイズの濃縮相をゲル中に固定し、濃縮相内でのアミロイド核形成の動態を調べた(図2A)。この研究では、酵母プリオンタンパク質である Sup35 NM ドメイン (Sup35NM)¹³⁾をモデルタンパク質とし、核形成速度(単位時間中に単位体積で起こる核形成イベントの数)の計測を目的とした。

2.2 実験条件

まず、アガロースゲル中に Sup35NM の濃縮相を生成・固定した。39 μM Sup35NM, 2% アガロース, 10% ポリエチレングリコール (PEG), 20 μM チオフラビン T (ThT), 4.3 mM リン酸緩衝液 (pH 6) となるように試薬類を 40 °C のマイクロチューブ中で混合することで濃縮相を形成した。その後サンプルを 384 ウェルプレートに移し、冷蔵庫 (4 °C) で 5 分間冷却することでアガロースをゲル化した。

アガロースゲル中に固定した Sup35NM 濃縮相を、共

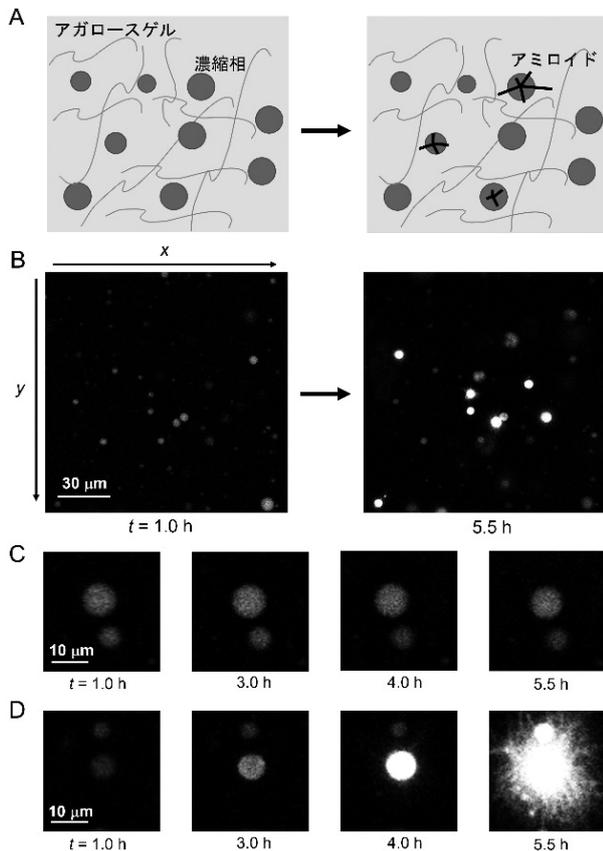


図2 アガロースゲル中に固定したタンパク質濃縮相におけるアミロイド形成

(A) アガロースゲル中に Sup35NM 濃縮相を固定している模式図。(B) 全体の蛍光画像の時間変化。(C) 核形成が起きなかった濃縮相の時間変化。(D) 核形成が起きた濃縮相の時間変化。Reprinted from ref 5. Copyright 2023 American Chemical Society.

焦点顕微鏡 (LSM 710, Carl Zeiss) を用いて観察した。ここでは、アミロイド線維伸長に伴い増強する ThT の蛍光を観察した。0.5~1 時間間隔で約 6 時間観察した。各時点で 1.19 mm² の領域を 3 μm 間隔で 10 枚の z-スライス画像を得た。画像解析の処理コストを削減するための工夫として、一つの画像データを 3×3 の九つのタイルに分割した。濃縮相の蛍光強度は時間とともに数十倍に増加するため、鮮明な画像を得るため三つのゲインを用いて計測した。つまり 1 回の実験でおおよそ 10 (時間数) × 10 (z-スライス数) × 9 (タイル数) × 3 (ゲイン数) = 2700 枚の画像を得た。

2.3 結果の概要

図 2B に示すように、時間とともに ThT 蛍光強度が上昇する濃縮相数が増えた。この挙動は、核生成が線維伸長に比べて非常に遅いため見られる。濃縮相中で核生成が 1 回も起こらないと図 2C のように濃縮相の蛍光強度は低いままであった。一方で、1 回以上核生成が起ると図 2D のように濃縮相内のタンパク質は速やかにアミロイド線維に変換し ThT 蛍光が上昇した。

3 Python を用いた画像解析スクリプト

3.1 本スクリプトの概要

本スクリプトの目的は、1 回の実験で得られた共焦点顕微画像 (図 2B) より、一つ一つの濃縮相の蛍光強度 (図 11B) およびサイズ (図 11C) の時間変化を明らかにすることである。この操作により蛍光強度が上昇した (= 核生成が起きた) 濃縮相の割合が求まり、濃縮相中のアミロイド核生成速度が算出できる。

上記目的を踏まえ、本スクリプトでは大まかに、まず各時間における画像を解析し (3.2~3.7)、その後すべての時間の情報を統合して、結果を出力した (3.8~3.9)。各項の要点は下記のとおりである。

- Python プログラミング環境の準備 (3.2)

Python をインストールし、画像解析に必要なライブラリ群をインポートした。

- 顕微画像の行列化 (3.3)

Python で画像解析をするために画像ファイルを行列として読み込んだ。

- 濃縮相の検出 (3.4)

画像を二値化して画像内の濃縮相を検出した。

- マスク処理 (3.5)

3.4 の結果を用いてマスクを作成し、各濃縮相の位置・面積情報等を取得した。

- Z 軸方向の濃縮相の情報統合 (3.6)

一つの濃縮相は z-スライス複数枚にまたがって検出された。そのため、解析における重複を避けるために濃縮相の重心位置を特定し、同一濃縮相の情報を統合した。同時に各濃縮相に ID を付与した。

- 濃縮相のデータ出力 (3.7)

各濃縮相の座標・蛍光強度・面積の情報を出力した。

- 全時刻・全タイルの濃縮相情報の統合 (3.8)

3.7 の出力データを基に各時刻における画像の位置ずれを補正し、全時刻・全タイルで統合した濃縮相 ID を付与した。

- 全時刻・全タイルの濃縮相データ出力 (3.9)

3.8 を踏まえ各濃縮相のすべての時刻における蛍光強度・面積情報を出力した。

次項からこれらの操作の詳細を記す。

3.2 Python プログラミング環境の準備

Python を使用する際には、Anaconda¹⁴⁾等のプラットフォームを利用するとよい。Anaconda には、Python のツールやライブラリがすでに含まれており、煩雑な環境構築の必要なく円滑に Python を使用できる。また、AnacondaNavigator¹⁵⁾を用いれば、新しい仮想環境の作成から内包されていないライブラリのインストールまで容易に実行できる。Python にはスクリプトモードとインタラクティブモードがあるが、本研究ではスクリプト

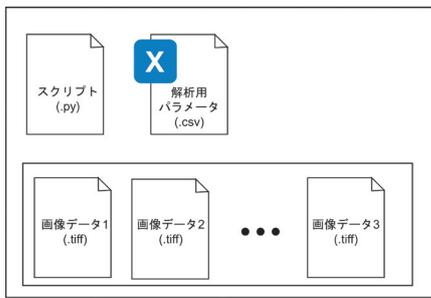


図3 ファイルの配置図

モードを用いた。

ファイルの配置を図3に示す。一つのフォルダの中にスクリプト、画像データを格納したフォルダ、実験名やゲイン情報など解析に必要なパラメータが記載されている csv ファイルを格納した。

そしてスクリプト冒頭で NumPy, pandas, os, glob および OpenCV を、import ステートメントを用いてスクリプトの最上部でインポートした¹⁶⁾。

3.3 顕微画像の行列化

画像は、ピクセルと呼ばれる小さな点の集合で構成されており、各ピクセルに色や明るさなどの情報が含まれている(図4)その集合体として画像が表現されるが、画像解析を行う際は、これらの情報を数値の行列として取り扱う。

まず、pandas ライブラリの read_csv 関数を用いて CSV ファイルから実験情報を読み取った。pandas ライブラリとはデータ操作用のライブラリであり¹⁷⁾、csv ファイルにおける表の操作の際に重宝する。

次に読み込んだ情報を基に全顕微画像のファイル名を取得した。ファイル名の取得には glob ライブラリの glob 関数を用いた。glob ライブラリとは、ファイルシステム内のファイルやディレクトリを検索するための標準ライブラリである¹⁸⁾。glob 関数は指定したパスを持つファイル名を一括で取得できる関数である。まず空のリストを作成し、この関数を用いて順次ファイル名をリストに格納した。

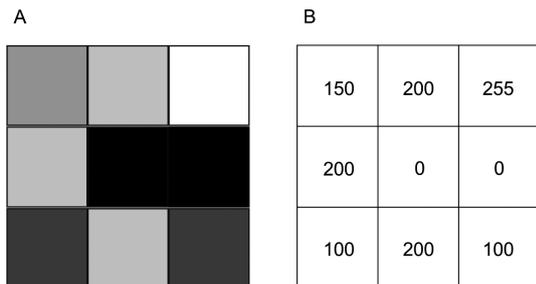


図4 画像の模式図

グレー画像はピクセルごとの色深度で記述できる。8 bit グレースケールの場合、色深度は0~255の範囲で表される

続いて、取得したファイル名の画像を、OpenCV の imread 関数を用いて読み込んだ。OpenCV とは画像解析用の汎用的なライブラリであり¹⁹⁾、imread は画像を行列(正確には NumPy 配列)として読み込む。NumPy とは科学技術計算を行う汎用的なライブラリである²⁰⁾。特に数列の操作・演算に長けているため、画像解析では、OpenCV と NumPy を組み合わせて使うことが多い。後の画像の解析手順を踏まえ、ここでは一つの画像について、四つの行列を用意した: 濃縮相の①検出用・②蛍光強度計測用の行列(それぞれ 16 bit グレースケール画像に対応) および③検出結果出力用・④蛍光強度測定結果出力用の行列(それぞれ RGB カラー画像に対応)を用意した。①, ②のグレースケール画像の読み込みでは、まず np.zeros 関数を用いて z-スタック数, ゲイン数, y 軸方向のピクセル数, x 軸方向のピクセル数の四次元のパラメータを持つすべての要素が0の初期行列を作成した。次いで各 z-スタック・ゲインに対応するファイル名を、先程作成した画像ファイル名のリストから、二重の for 構文と imread 関数を用いて行列として取り込んだ。③④の画像についてもおおむね同様の手順で行列を作成した。③④の画像では RGB の3チャンネルの数値を入力する必要があるため、①②の読み込みで準備した行列に次元を一つ追加し5次元の行列を作成した。

以上の手法で取得した行列を用いて次項以降の解析を行った。

3.4 濃縮相の検出

本項では、画像中の濃縮相の輪郭を検出し、行列として出力する方法について説明する。

まず、OpenCV ライブラリの threshold 関数を用いて、3.3 で読み込んだ①検出用行列を二値化処理した。二値化のタイプには cv2.THRESH_BINARY を使用し、^{しきい値} 閾値より大きい値は 255 (8 bit 出力での真っ白に相当する)、閾値以下の値は 0 (同真っ黒に相当する) に変換した。

次に、OpenCV ライブラリの findContours 関数を用いて二値化した行列から輪郭(白と黒の境目)を検出した。第2引数としては cv2.RETR_TREE を使い、輪郭の階層構造をツリー形式で取得した。また、輪郭座標には第3引数として cv2.CHAIN_APPROX_SIMPLE を用いた。

その後、画像上のノイズ(ごく小さな白い輝点)を除去するために、OpenCV ライブラリの contourArea 関数によって 20 pixel² 以下の面積のオブジェクトを削除した。

以上で得られた輪郭情報を、OpenCV ライブラリの drawContours 関数を用いて③検出結果出力用行列と④蛍光強度出力用行列に追記した。そして OpenCV ライブラリの imwrite 関数を用いてこれらの行列を画像フ

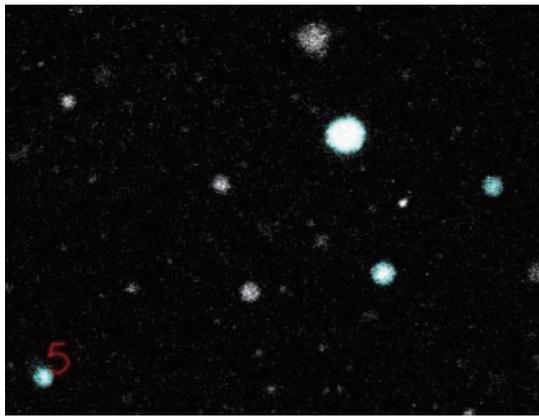


図5 濃縮相の輪郭を検出した例

蛍光強度や面積が一定以下のシグナルは棄却された。Reprinted from ref 5. Copyright 2023 American Chemical Society.

イルとして出力した(図5)。ここでは画像ファイルの軽量化のためTIFFではなくJPEGで出力した。また3.5のマスク作成の準備として、NumPyライブラリのzeros_like関数で真っ黒な画像に相当する行列を準備し、そこに輪郭情報を記入した(マスク用行列)。

3.5 マスク処理

マスク処理はOpenCVライブラリのconnectedComponentsWithStat関数で行った。本関数では二値化された画像中に含まれるすべてのオブジェクトを検出し、それらをラベリングし、面積、重心座標を検出する。そして、図6のようにピクセルごとにラベリングした結果を出力する。注意点として、マスク処理は背景全体を「0番目のオブジェクト」として認識するので、実際に利用する際は、オブジェクト数や重心から0番目のものを除去する必要がある。

3.4で準備したマスク用行列に対しconnectedComponentsWithStat関数を用いて、合計のオブジェクト数および各オブジェクトのマスク番号・面積、重心座標を出力した。この操作をすべてのマスク行列に対して

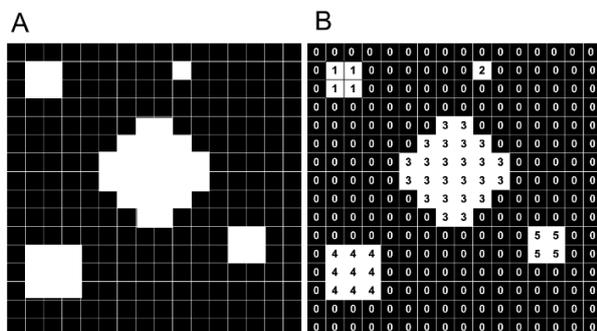


図6 マスク処理の概要図

(A) 二値化した画像(白色部分が濃縮相を表す)。(B) マスク処理後の画像。

行い、結果を行列に格納した。この処理により、画像中の濃縮相一つ一つに対する議論が可能になった。

3.6 z軸方向の濃縮相の情報統合

本実験では、濃縮相が数 μm 程度の大きさであり、 $3\mu\text{m}$ 間隔でz-スライス画像を取得しているため、同一の濃縮相が複数の画像に出現する(図7)。本項では、複数の画像に渡って現れる同一濃縮相の情報を統合する方法について説明する。

本操作の概要を説明する。まず、z軸方向にスライスした複数の画像において、濃縮相の重心座標が一致しているものは同じ濃縮相であると考えた。そして、共焦点顕微鏡は焦点距離前後の蛍光もある範囲で拾うため、濃縮相の中心付近のz画像ほど、蛍光強度は大きくなる(図7)。そのため、同一濃縮相の各z-スライス画像について重心付近の蛍光強度を比較し、最大となるz-スライス画像が濃縮相の中心に最も近いz画像であると判断した。以上の操作で得られたz座標、xy中心座標、マスク番号などの情報を行列として出力した。

本解析について、より詳細に説明する。各ファイルの最小ゲイン画像における濃縮相を、for文とif文を組み合わせることで次の①~③の選別にかけた。

①注目している濃縮相が、既に検出されたものと同一でないか確認する。そのため、一つ目の濃縮相が検出されるまでは、この選別は実質的に飛ばされる。そして既に検出された濃縮相との重心の座標の差がx座標、y座標ともに4ピクセル以内である場合は、既に検出された濃縮相と同一の濃縮相であると判断し棄却した。それ以上離れている場合は新しい濃縮相であると判断して②のプロセスに進み、さらなるふるいにかけた。

②濃縮相の中心に最も近いz-スライス画像を特定する。そのため、検知している濃縮相のxy中心座標におけるの蛍光強度が、上下のz-スライス画像における同蛍光強度よりも大きい場合のみ、濃縮相の中心を捉えられていると判断して③に進んだ。蛍光強度が極大となっていない場合は、他のz-スライス画像上に中心があるため、ここでは棄却した。

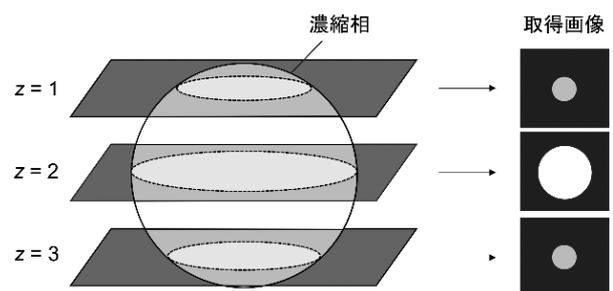


図7 z-スライスと取得画像のイメージ

一つの濃縮相が複数のz-スライス画像で検出される場合、中心付近の画像(図の $z=2$)で蛍光強度と面積が最大となる。

③蛍光強度が最小ゲインにおいて飽和しているピクセルが存在する場合には、蛍光強度を定量的に議論できないため棄却した。

以上の選別を突破した濃縮相を「蛍光計測が可能な新規濃縮相」だと考え、ID 番号を付与した。そして ID、検知された z-スライス番号、xy 座標、マスク番号等を行列に格納した。この操作により、複数の z-スライス画像に渡って存在していたそれぞれの濃縮相の情報が一つに統合され、正しい濃縮相数の判定が行えるようになった。

3.7 濃縮相の面積補正と解析データの出力

アミロイドが形成されると、濃縮相の外まで伸長することで濃縮相外にも蛍光が観測され、実際の濃縮相の大きさや重心座標が計測できない(たとえば図 2D 5.5 h)。アミロイドが密に存在する濃縮相内と比較して、アミロイドが伸びて濃縮相外にはみ出た部分の蛍光強度は小さくなる。このことから、濃縮相内最大蛍光の半分以上の強度を持つ範囲が濃縮相であると判定し、濃縮相からはみ出たアミロイドによる面積の誤認識を軽減した。

スクリプト上の操作としては、まず、ある濃縮相について、内部の蛍光強度が飽和しないゲインの画像を用いて最大蛍光強度を求めた。次に、if 構文により濃縮相内の各ピクセルの蛍光強度が最大蛍光強度の 0.5 倍の値より大きいか判定し、その x 座標と y 座標をそれぞれ加算した。その後、判定された点の数(真の濃縮相面積に相当)でそれらを割ることにより、濃縮相の真の重心座標と面積を求めた(図 8)。

以上の手順で求めた真の重心座標や面積、さらに濃縮相の蛍光強度を、3.6 と同じ行列に格納した。これにより、ある時刻・あるタイルで観測された全濃縮相(～40 個)の重心座標や蛍光強度、面積、マスク番号などの詳細な情報を一つの行列に統合することができた(図 9)。



図 8 濃縮相の正確な大きさを検出した例

3.4 の操作では水色で囲まれた部分が検出されたが、3.7 の操作により薄い水色で囲まれた強度の高い中心部分のみが濃縮相として検出された。

濃縮相番号	zスライス番号	マスク番号	x座標	y座標	面積	ゲイン	蛍光強度
0	1	11	870.7	65.4	32	1.00E+06	4.88E+05
1	1	24	449.4	287.4	363	2.01E+05	5.85E+06
2	1	27	677.1	285.5	214	2.01E+05	5.01E+06
3	1	30	900.6	290.3	54	2.01E+05	7.32E+05
4	2	36	96.8	369.4	119	2.01E+05	1.92E+06
5	2	43	567.1	410.4	33	1.00E+06	5.97E+05
6	2	48	608.1	445.5	36	1.00E+06	1.64E+06
7	2	54	306.7	559.0	71	2.01E+05	5.68E+05
8	3	56	584.3	511.2	55	2.01E+05	9.60E+05

図 9 出力された濃縮相情報のイメージ

3.8 全時刻・全タイルの濃縮相情報の統合

本項では、3.7 で出力された時刻・タイルごとで出力されている濃縮相情報を、実験全体を通して統合する方法について説明する。

本操作は三段階で行った。①時間が経つにつれて濃縮相の位置が徐々に移動するため、そのずれを補正した。②同一濃縮相に対して全時刻を通して同一の ID を付与した。③全タイルについてすべての濃縮相に対して固有の ID を振りなおした。

まず、①のずれ補正について説明する。図 10A に一般的な時間変化画像の模式図を示す。本研究の観察法では、時刻 $t=0$ と $t=t_n$ の画像比べると濃縮相の位置が数 μm ずれてしまう。そのため、 $t=t_n$ の画像を 1 ピクセルずつ動かし、このずれを補正した。

具体的には時刻 $t=t_n$ の全濃縮相の座標データにおいて、x, y 座標を $\Delta x, \Delta y$ ずつずらした。そして、 $t=0$ のすべての濃縮相の xy 座標データと比較し、差が 5 pixel 以内に収まっている濃縮相を「ずれを補正できた濃縮相」と定義して数え上げた(図 10B)。 $-20 < \Delta x < 20$, $-20 < \Delta y < 20$ の範囲でこの操作を実行し、ずれを補正できた濃縮相数が最も多かった $(\Delta x, \Delta y)$ を求め、これを、 $t=t_n$ でのずれの補正值とした。以上の操作を全時刻・全タイルで行い、濃縮相の位置のずれを補正した。

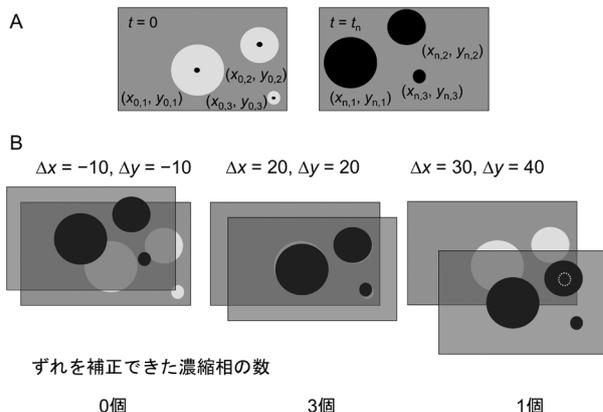


図 10 時間による画像ずれの補正方法

(A) 異なる時刻における濃縮相のずれの模式図。(B) $t=0$ を基準として、重なった(=ずれを補正できた)濃縮相が最も多くなる x, y 方向の移動度 $(\Delta x, \Delta y)$ を決めた。

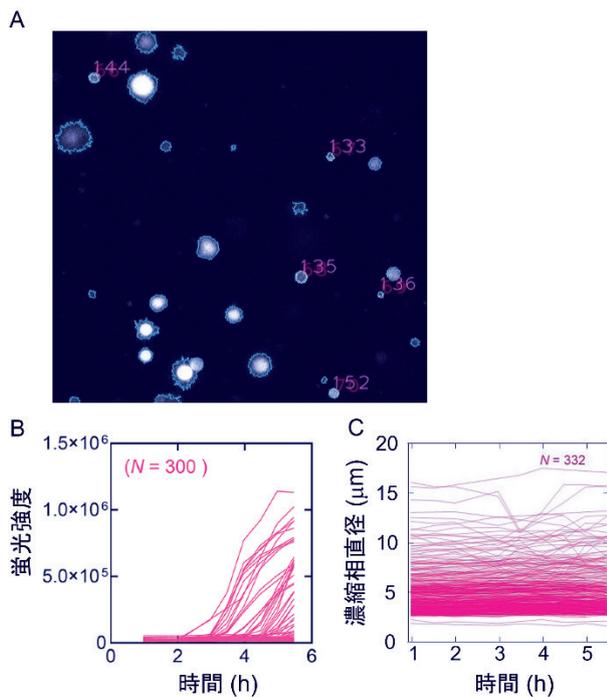


図 11 画像解析の結果

(A) 濃縮相が統合された画像. (B) 各濃縮相の蛍光強度の時間変化. (C) 濃縮相直径の時間変化. Reprinted from ref 5. Copyright 2023 American Chemical Society.

また、この操作は計算コストが大きいため、画像を9分割してコストを下げる、都度補正後の結果を保存するなどの工夫が必要であった。

次に、②同一濃縮相に対して全時刻において同一IDを付与した。具体的には、①の後、 $t=0$, t_n においてタイル・xyz座標が同じ濃縮相を同一ものと考え同一IDを付与した。また、一度しか登場しない濃縮相情報は削除した。

最後に、③九つすべてのタイルの全濃縮相について固有のIDを付与した。全タイルの濃縮相情報を、NumPyライブラリのunique関数を用いて重複を避けながら、一つのリストにまとめていった。その後、濃縮相を1から順にIDを振り直すことですべての濃縮相に固有のIDを付与できた(図11A)。

3.9 全タイル・全時刻の濃縮相データ出力

3.8において濃縮相IDが統一できたため、これをもとに面積と蛍光強度の時間変化を濃縮相一つ一つについてまとめた。

この際、ゲインが異なると蛍光強度を直接比較できないため、各ゲインにおいて補正値を用いて蛍光強度を補正した。また、ある時刻の濃縮相の面積が $t=0$ における面積の2倍以上となっている場合は、適切に濃縮相を測定できていないと判断してデータを棄却した。

以上により、それぞれの濃縮相の各時間における蛍光強度(図11B)と直径(濃縮相を円形と近似し導出。図

11C)の情報を統合できた。最後にこの行列をNumpyライブラリのsavetxt関数でcsvファイルとして出力し、エクセルなどを使ってグラフ作成等を行った。

4 おわりに

本稿では、筆者らの過去の実験データを基にPythonを用いた3次元共焦点顕微鏡画像の解析の具体的な方法について概説した。画像解析において必要となるライブラリについて言及し、マスク処理など基本的な画像処理について説明した。

本研究では、ごく単純なアミロイド生成実験系において、大量(2700枚)の画像を独自スクリプトにて解析することで、核生成についての新しい知見が得られた。スクリプト作成が分析化学分野の実験的研究において強力なツールとなり得ることが分かる。一方で、本研究で著者らは高品質な実験データの重要性を再認識した。人間の目では簡単に判別できるような差であっても、スクリプトで検出できないことが多々あった。(そのため本研究では多くのゲインで撮影する必要があり計算コストが激増した。その割に得られたデータ点数が少なく、解析誤差が大きかった。)そのため、実験ベースの分析化学手法の進歩もこれまでと同様に重要であることは間違えない。

本稿が自身の研究に画像解析を実装しようと考えている研究者の一助となることを期待する。

謝辞 本研究は科研費(22K18316)、創発的研究支援事業(JPMJFR211Y)、内藤記念女性研究者研究助成金文部科学省先端研究基盤共用促進事業(JPMXS0440600021&22)の支援を受けたものである。

文 献

- 1) T. Yasui, T. Yanagida, S. Ito, Y. Konakade, D. Takeshita, T. Naganawa, K. Nagashima, T. Shimada, N. Kaji, Y. Nakamura, I. A. Thiodoru, Y. He, S. Rahong, M. Kanai, H. Yukawa, T. Ochiya, T. Kawai, Y. Baba : *Science Advances*, **3**, e1701133 (2017).
- 2) X. Lyu, V. Hamedpour, Y. Sasaki, Z. Zhang, T. Minami : *Analytical Chemistry*, **93**, 1179 (2020).
- 3) S. Tomita, H. Kusada, N. Kojima, S. Ishihara, K. Miyazaki, H. Tamaki, R. Kurita : *Chemical Science*, **13**, 5830 (2022).
- 4) <https://www.python.org/>, (2024年7月12日, 最終確認).
- 5) M. Fukuyama, S. Nishinami, Y. Maruyama, T. Ozawa, S. Tomita, Y. Ohashi, M. Kasuya, M. Gen, E. Chatani, K. Shiraki, A. Hibara : *Anal. Chem.*, **95**, 9855 (2023).
- 6) <https://www.w3schools.com/python/default.asp>, (2024年7月12日, 最終確認).
- 7) M. R. Sawaya, M. P. Hughes, J. A. Rodriguez, R. Riek, D. S. Eisenberg : *Cell*, **184**, 4857 (2021).
- 8) T. P. J. Knowles, M. Vendruscolo, C. M. Dobson : *Nat. Rev. Mol. Cell Biol.*, **15**, 384 (2014).
- 9) R. Crespo, F. A. Rocha, A. M. Damas, P. M. Martins : *Journal of Biological Chemistry*, **287**, 30585 (2012).
- 10) J. T. Joseph, L. T. Peter : *Cell*, **73**, 1055 (1993).

- 11) S. Mukherjee, M. Poudyal, K. Dave, P. Kadu, S. K. Maji : *Chem. Soc. Rev.*, **53**, 4976 (2024).
- 12) S. Ray, N. Singh, R. Kumar, K. Patel, S. Pandey, D. Datta, J. Mahato, R. Panigrahi, A. Navalkar, S. Mehra, L. Gadhe, D. Chatterjee, A. S. Sawner, S. Maiti, S. Bhatia, J. A. Gerez, A. Chowdhury, A. Kumar, R. Padinhateeri, R. Riek, G. Krishnamoorthy, S. Maji : *Nature Chemistry*, **12**, 705 (2020).
- 13) T. M. Franzmann, M. Jahnel, A. Pozniakovsky, J. Mahamid, A. S. Holehouse, E. Nüske, D. Richter, W. Baumeister, S. W. Grill, R. V. Pappu, A. A. Hyman, S. Alberti : *Science*, **359**, 6371 (2018).
- 14) <https://www.anaconda.com/>, (2024年7月12日, 最終確認).
- 15) <https://docs.anaconda.com/navigator/>, (2024年7月12日, 最終確認).
- 16) <https://docs.python.org/ja/3/reference/import.html>, (2024年7月12日, 最終確認).
- 17) <https://pandas.pydata.org/>, (2024年7月12日, 最終確認).
- 18) <https://github.com/python/cpython/blob/3.12/Lib/glob.py>, (2024年7月12日, 最終確認).
- 19) <https://OpenCV.org/>, (2024年7月12日, 最終確認).
- 20) <https://NumPy.org/ja/>, (2024年7月12日, 最終確認).



小林 恒一 (KOBAYASHI Koichi)
 東北大学多元物質科学研究所 (〒980-8577 宮城県仙台市青葉区片平 2-1-1).
 東北大学理学部化学科卒業. 学士 (理学).
 《現在の研究テーマ》細胞内における蛍光性タンパク質結晶核の生成速度解析. 《趣味》フットサル.



山内 皓太 (YAMAUCHI Kota)
 東北大学多元物質科学研究所. (〒980-8577 宮城県仙台市青葉区片平 2-1-1).
 東北大学理学部化学科卒業. 学士 (理学).
 《現在の研究テーマ》ナノメートルスケールの濃縮相を用いたアミロイド核形成速度の解析. 《趣味》ラジオを聴くこと.



福山 真央 (FUKUYAMA Mao)
 東北大学多元物質科学研究所. (〒980-8577 宮城県仙台市青葉区片平 2-1-1).
 東京大学工学系研究科応用化学専攻博士課程修了. 博士 (工学). 《現在の研究テーマ》マイクロ流体を用いた液液・固液界面現象の解析と微量分析応用. 《趣味》鉱物磨き.

原 稿 募 集

トピックス欄の原稿を募集しています

内容：読者の関心をひくような新しい分析化学・分析技術の研究を短くまとめたもの.

執筆上の注意：1) 1000字以内 (図は1枚500字に換算) とする. 2) 新分析法の説明には簡単な原理図などを積極的に採り入れる. 3) 中心となる文献は原則として2年以内のものとし, 出所を明記する.

なお, 執筆者自身の文献を主として紹介するこ

とは御遠慮ください. 又, 二重投稿は避けてください.

◇採用の可否は編集委員会にご一任ください. 原稿の送付および問い合わせは下記へお願いします.

〒141-0031 東京都品川区西五反田 1-26-2
 五反田サンハイツ 304号

(公社)日本分析化学会「ぶんせき」編集委員会
 [E-mail : bunseki@jsac.or.jp]