Simulating FRAP: An Exploration of Diffusion Models

Grace Zhou

18 December 2023

Introduction and Background

Imaging techniques are key to the study of living processes, and the advent of fluorescent proteins has proved crucial to the rapid development of increasingly sophisticated live-cell imaging techniques. A particular advantage of fluorescent proteins is that their fluorescence is genetically encoded (within the primary amino acid sequence): thus the development and consequent refinement of green fluorescent proteins and their contemporaries has permitted the practical usage of powerful fluorescent imaging in labs [1].

Techniques used for real-time study include FRAP, short for fluorescence recovery after photobleaching, which can be used to study protein dynamics in cells among other diffusion-based phenomena. Photobleaching is a phenomenon that stems from exposing fluorophores to high-intensity illumination, altering their photochemical structure such that they can no longer fluoresce via the excitation of electrons [2]. FRAP then works via a combination of photobleaching and analysis of molecules tagged with fluorescent proteins: a high intensity laser pulse is used to irreversibly bleach a targeted subcellular region of tagged molecules, and we expect the fluorescence to return to some equilibrium state due to the diffusion of bleached and unbleached molecules [2]. Quantitative measurement of that recovery is typically done by graphing time against normalized intensity. Upon reaching equilibrium, the curve should plateau at a certain intensity at a certain time: fitting an equation to the curve then allows us to measure the effective diffusion coefficient of the medium [3]. A schematic of a recovery curve as well as the associated FRAP process is shown in Figure 1.

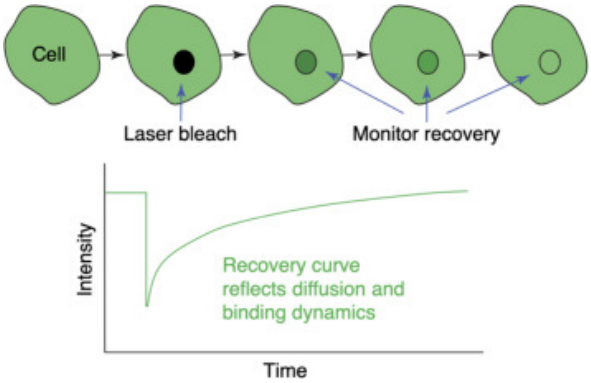


Figure 1*: FRAP laser bleaching and recovery curve schematic.*

FRAP is used primarily to analyze the diffusion of proteins in cells, but its versatility allows for frequent use in a variety of applications, including the analysis of protein dynamics in cell membranes (following the maturation of cell membrane models to include embedded proteins) [2]. More recent advances in the technology have also permitted the analysis of protein binding as well as diffusion [4]. Still, understanding the diffusion of proteins has significant pharmacological implications, especially with regards to drug delivery and measuring drug diffusion through body tissues both *in vitro* and non-invasively *in vivo.* In particular, FRAP's ability to detect and analyze diffusion even in more complex materials is of special utility in designing time-controlled drug delivery systems in synthetically engineered hydrogels, the comparative efficacy of which can be measured via FRAP technology [5].

As a relatively new technique, FRAP is continually being refined and adjusted on multiple fronts, including in the development of more photochemically stable GFPs and better mathematical models that account for the complexities of molecular dynamics. Classical FRAP, moreover, is relatively simple to perform and requires only a microscope set-up involving a stationary laser beam, which produces a circular photobleached region of approximately uniform intensity (Figure 1); later iterations with confocal laser scanning microscopes (CLSM) have allowed the photobleaching of more arbitrary patterns that better account for fluorophore sensitivity and noise [5]. As such, we continue to see developments both in how we perform FRAP and how we analyze the data it produces.

The Model: Results and Discussion

We split FRAP into two stages: the photobleaching stage and the post-photobleach stage. Stochastic simulations and deterministic models are developed for both stages.

The photobleaching stage is represented by the decay of a single chemical species, 'A':

$$\mathrm{A} \rightarrow \mathrm{B},$$

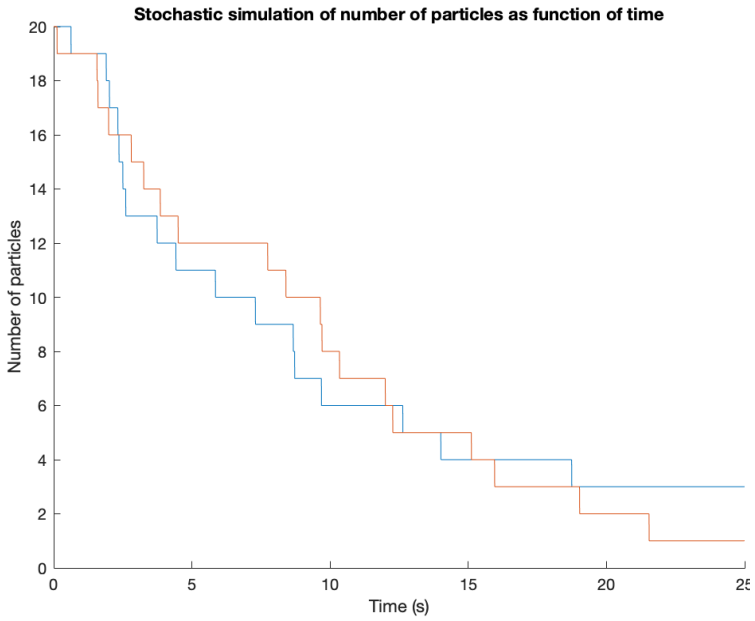
which we suppose has reaction constant k. Species B (which A degrades into) is of little interest here: we would simply like to simulate the concentration of A over time, where A's concentration is analogous to the intensity of the photobleached region. By the reaction kinetics of a first order reaction, k is defined such that $k\,\mathrm{d}t$ is the probability a random molecule of A degrades during the infinitesimally small time interval between t and $t + \mathrm{d}t$; it follows that if we denote the concentration

of molecules of A at time t to be A(t), then the probability the reaction occurs is A(t)k dt [6]. It follows that we can form a naïve stochastic simulation of the process through the following algorithm, taken from Erban [6]:

(1) Generating a random number r uniformly distributed in (0, 1)

(2) If $r < A(t)k\,\Delta t$, then set $A(t + \Delta t) = A(t) - 1$; otherwise, $A(t + \Delta t) = A(t)$. Then continue with step (1) for time $t + \Delta t$.

We then plot A(t) against time to produce a random walk through the degradation of A, as seen in Figures 2(a) and 2(b).



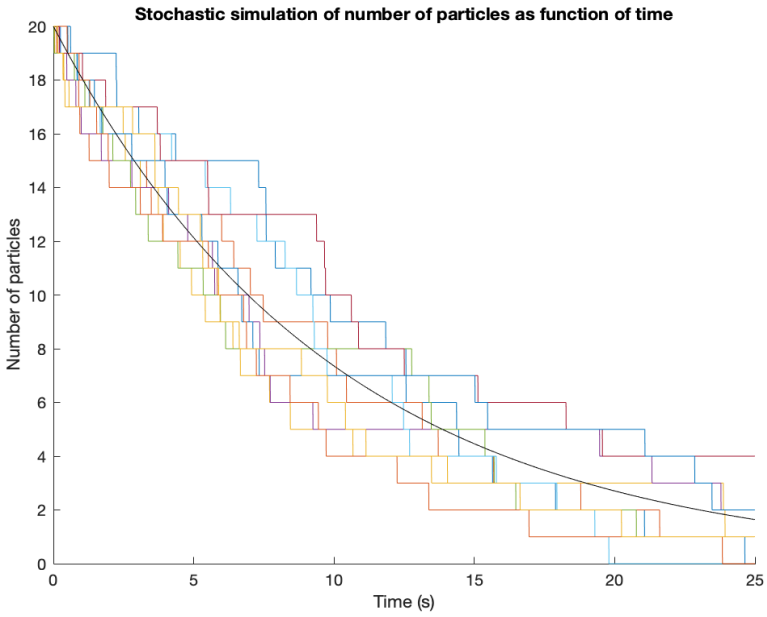


Figure 2(a), left*: 2 realizations of random walk with $k = 0.1\ s^{-1}$, A(0) = 20 and $\Delta t = 0.005$ s.* 2(b), right*: 10 realizations of random walk with $k = 0.1\ s^{-1}$, A(0) = 20 and $\Delta t = 0.005$ s. From the differential equation given by first order reaction kinetics, we expect that the deterministic trajectory of the particle concentration follows $A(t) = 20e^{-kt}$ (graphed as black curve), which matches the stochastic simulation mean.*

A more realistic stochastic simulation of the same reaction can be calculated by computing the precise time $t + \tau$ at which the next reaction ought to take place (as opposed to taking a fixed interval and randomly generating whether the reaction occurs or not). Following the computational process detailed in [6] gives

$$\tau = \frac{1}{A(t)k} \ln\left[\frac{1}{r}\right],$$

from which our stochastic algorithm follows:

(1) Generate a random number r uniformly distributed in (0, 1).

(2) Compute the time when the next reaction occurs as $t + \tau$, where τ is given by the equation above.

(3) The number of molecules at time $t + \tau$ is given by $A(t + \tau) = A(t) - 1$. Then continue with step (1) for time $t + \tau$,

where steps (1)-(3) are repeated until $A(t) = 0$, i.e. there are no molecules of A left [6]. The stochastic simulation that results is shown in Figure 3.

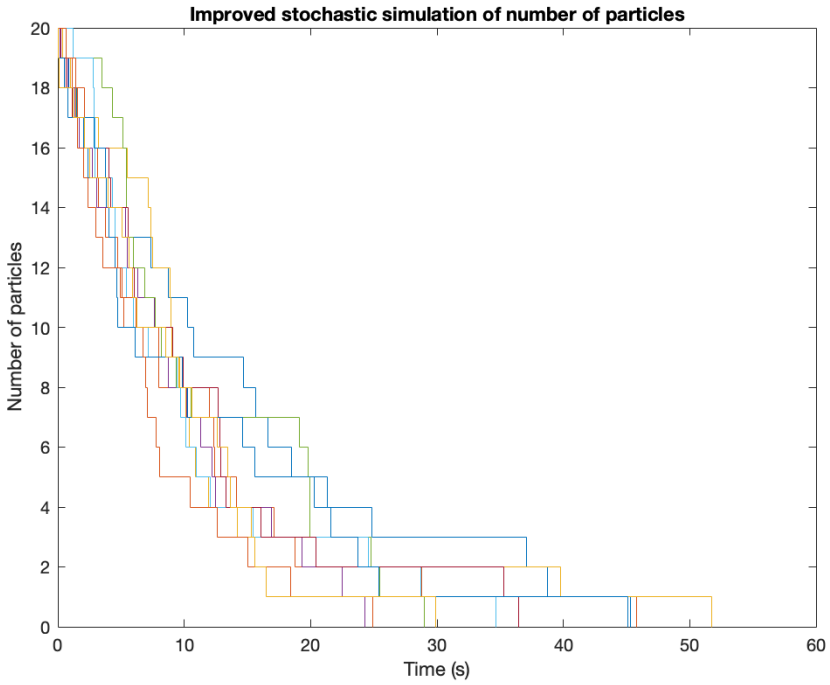


Figure 3: *10 realizations of the improved random walk with $k = 0.1\ s^{-1}$, $A(0) = 20$. This figure again aligns closely with the deterministic exponential equation, as expected: while it is highly unlikely that one stochastic realization overlaps fully with another, the general trend described by each realization follows what we would expect from the ODE of a single-order degradation. While the accuracy of this model is not immediately apparent when compared qualitatively to the previous model, the process of its derivation (Appendix 2) requires no approximation and therefore is more computationally intense when compared to the model in Figure 2, which loses accuracy as we decrease Δt.*

The post-photobleach stage in its simplest form imitates two-dimensional pure diffusion, as we suppose that the unbleached and bleached molecules diffuse in continuous space with no reflective boundaries or binding conditions. Moreover, we begin both our deterministic and stochastic simulations with a uniformly bleached circle at the center of our square "cell".

The deterministic equation for two-dimensional diffusion is the partial differential equation which is derived by removing the drift term from the Fokker-Planck equation (note that D is the

$$\frac{\partial U}{\partial t} = D\left(\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2}\right)$$

diffusion coefficient and U is the probability density function of the particle) [6, 7]. By applying finite-differences method and discretizing U (see Appendix 2 and [7]), we can set our initial boundary conditions to simulate the time evolution of U: we define our cell to be 1 mm by 1 mm with a photobleached circle of radius 0.2 mm at its center, setting our initial U such that the probability density function is uniformly maximized everywhere except for within the circle (where the density function is 0). Allowing U to evolve over time as dictated by the equation above then gives Figure 4. Note that in order to appropriately simulate a cell with non-absorbent borders, we set the boundaries of the cell to have maximal probability density per time-step, which ensures that all particles originally within the cell remain within the cell.

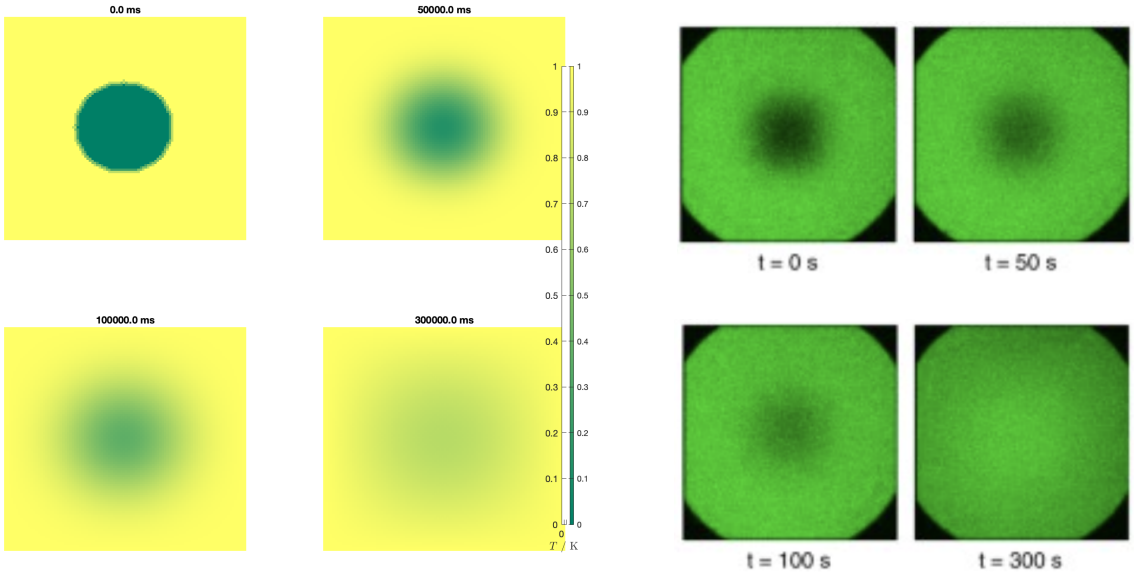


Figure 4: *Side-by-side of 300 second timeframe of simulated FRAP process, taking D = 1*10^{-4} mm^2 s^{-1} (left) and real FRAP process on fluorescein-dextran 70 kDA, which has similar D (right, taken from*

[8]). Note that maximum concentration of bleached particles is dark green on the simulated FRAP, as noted on the colorbar (the analogous region in the actual FRAP images is black). As expected, the two are qualitatively very close, and the simulation on the left behaves as expected: the photobleached molecules spread fully throughout our "cell" over time.

A more quantitative analysis of our deterministic FRAP model is done by taking the recovery curve, which we produce by graphing time against intensity. Since we have no way of directly measuring intensity here, we take it as a function of probability density, i.e. the total density of the photobleached region divided by the total possible density of the photobleached region (i.e. if the photobleached region was full of particles). The curve that results is shown in Figure 5.

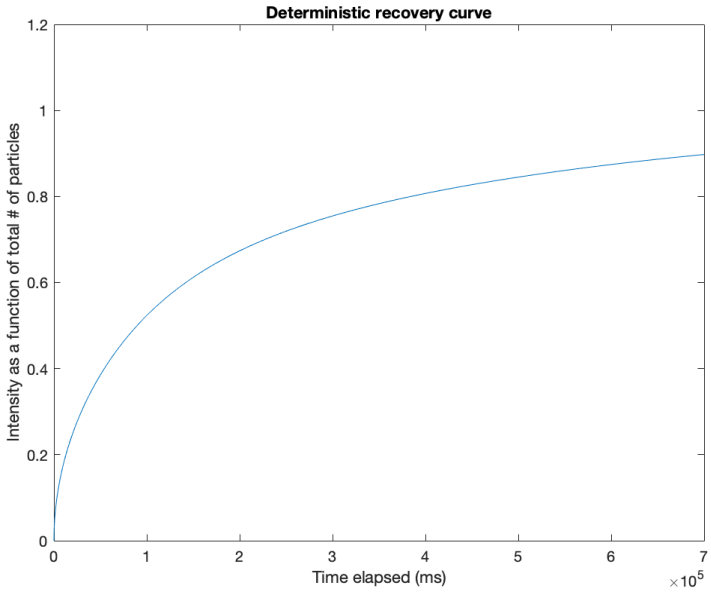


Figure 5: *Deterministic recovery curve for $D = 1*10^{-4}$ mm^2 s^{-1}. We see that the curve begins to plateau at an intensity of about 0.9, which matches what we expect from a recovery curve given that the fluorescence will eventually reach equilibrium.*

To form the proper equations for our stochastic simulation, we adapt the Fokker-Planck equation (recall the deterministic model–we continue to ignore drift) into a set of two stochastic differential equations (SDEs), the discretized versions of which are below:

$$X(t+\Delta t) = X(t) + \sqrt{2D\,\Delta t}\;\xi_x,$$
$$Y(t+\Delta t) = Y(t) + \sqrt{2D\,\Delta t}\;\xi_y,$$

from [6]. While the precise mathematical derivation of these equations is outside the scope of this paper, the derivation of SDEs generally involves the addition of a noise term, which is here represented by ξ. Note that the Brownian noise term is Gaussian and temperature-dependent [9]. Each particle in the stochastic simulation then performs a random walk, parametrized as follows:

(1) Generate two normally distributed random numbers ξ_x and ξ_y, with zero mean and unit variance.

(2) Walk them according to the SDEs listed above. Then continue with step (1) for time $t + \Delta t$.

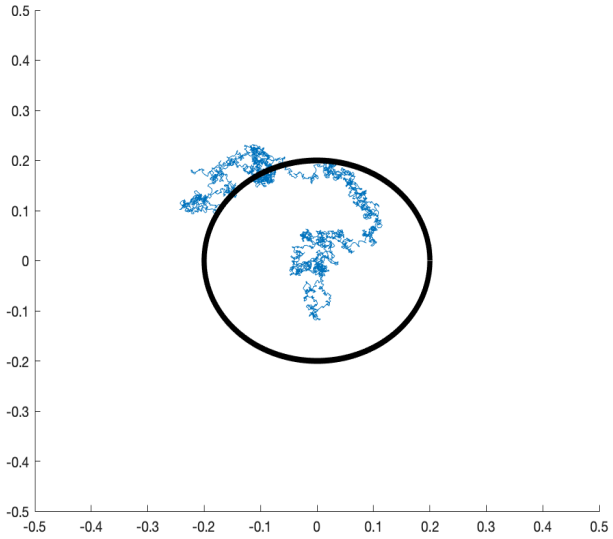


Figure 6: *example of random walk from the origin*.

The initial boundary conditions used for the deterministic model still hold in our stochastic simulation: we establish an 80 by 80 coordinate grid on a continuous space 1 mm by 1 mm cell. We draw a circle of radius 0.2 mm centered at the origin (the center of the cell) and identify all points outside of that circle, which are still fluorescent post-photobleach: we then random-walk them as above. A reflective barrier is imposed at the boundaries of our cell. Doing this over a time-frame of 300 seconds and taking snapshots of all points produces Figure 7.

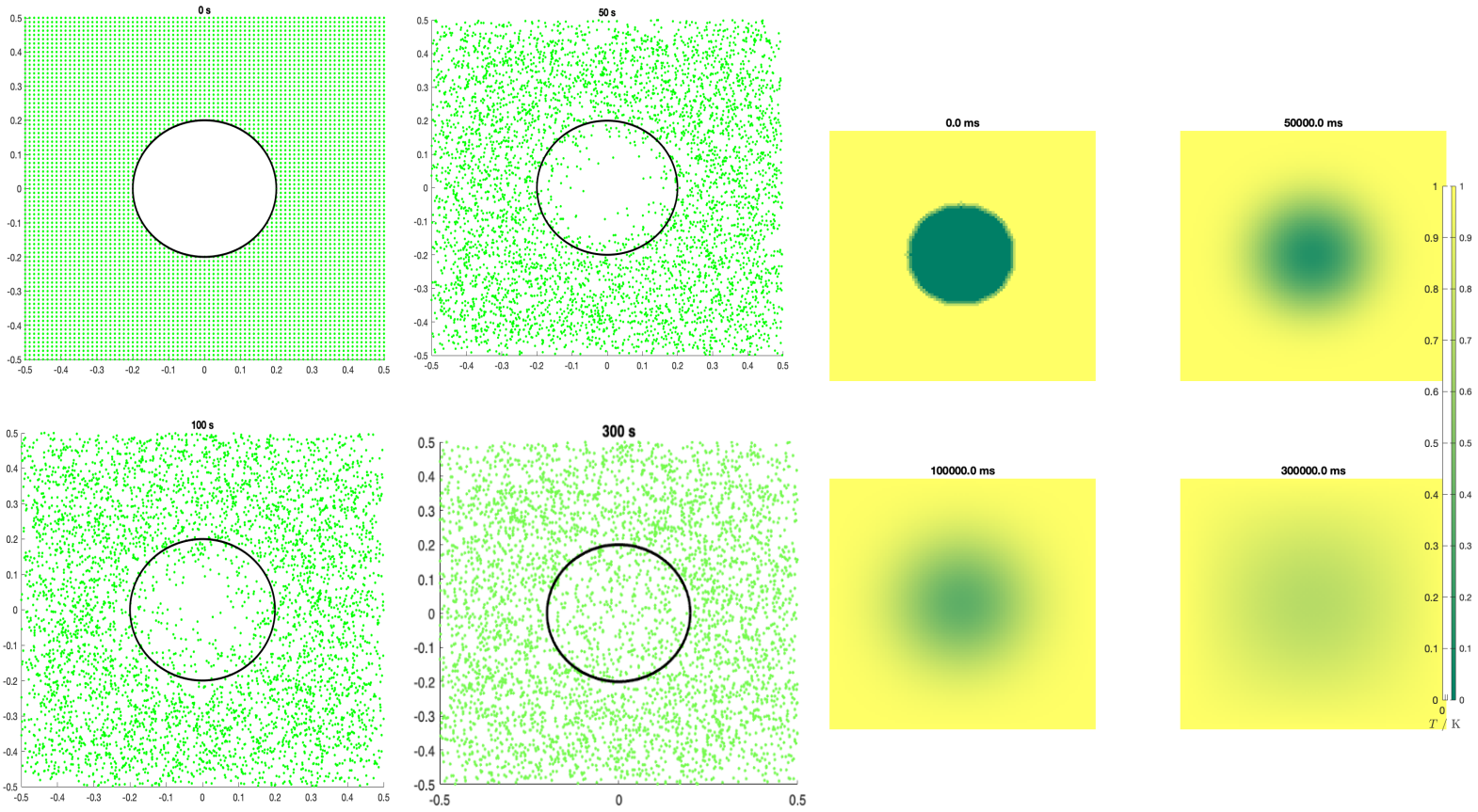


Figure 7: *progression of diffusion over time with radius 0.2mm and diffusion coefficient 1*10^{-4} mm^2 s^{-1}. As expected, we see a gradual spread of particles over the photobleached region. Our deterministic model from figure 4 is placed to the right for reference: we see that the two models are very similar per timeframe, as the gradient "fringe" in our deterministic model corresponds with the gradual entry of particles into the photobleached circle in the stochastic simulation.*

To obtain a recovery curve for the stochastic simulation, we once again graph time versus intensity, where intensity is measured as a function of some external factor. Here, we take intensity to be the number of particles inside the photobleached region divided by the area of the region. Some stochastic recovery curves are presented in Figures 8 and 9.

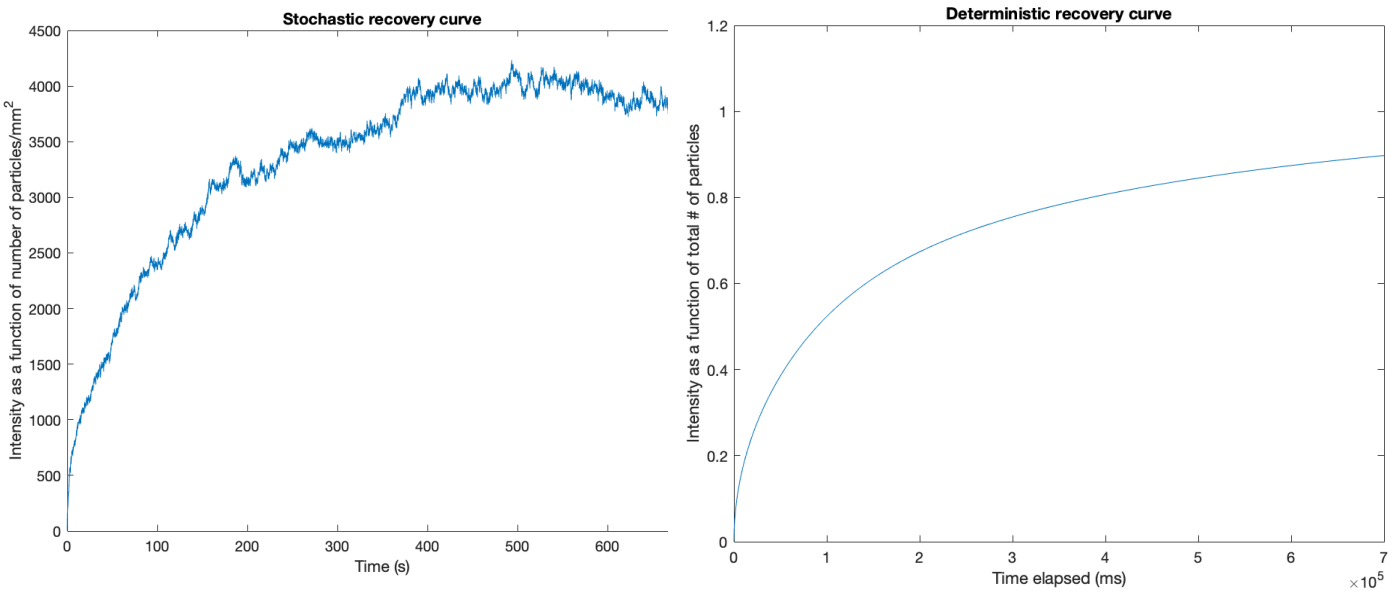


Figure 8: *Left: stochastic recovery curve given r = 0.2mm and D = 1*10^{-4} mm^2 s^{-1}. The deterministic recovery curve under the same conditions (figure 5) has been pasted at right over the same timeframe. Note that as expected, the two are very similar: while the stochastic curve is noisier given the random nature of its generation, both curves plateau at approximately similar times. Because the intensity is calculated differently in each simulation, the y-axes are scaled differently, but the approximate ratio of growth remains about the same.*

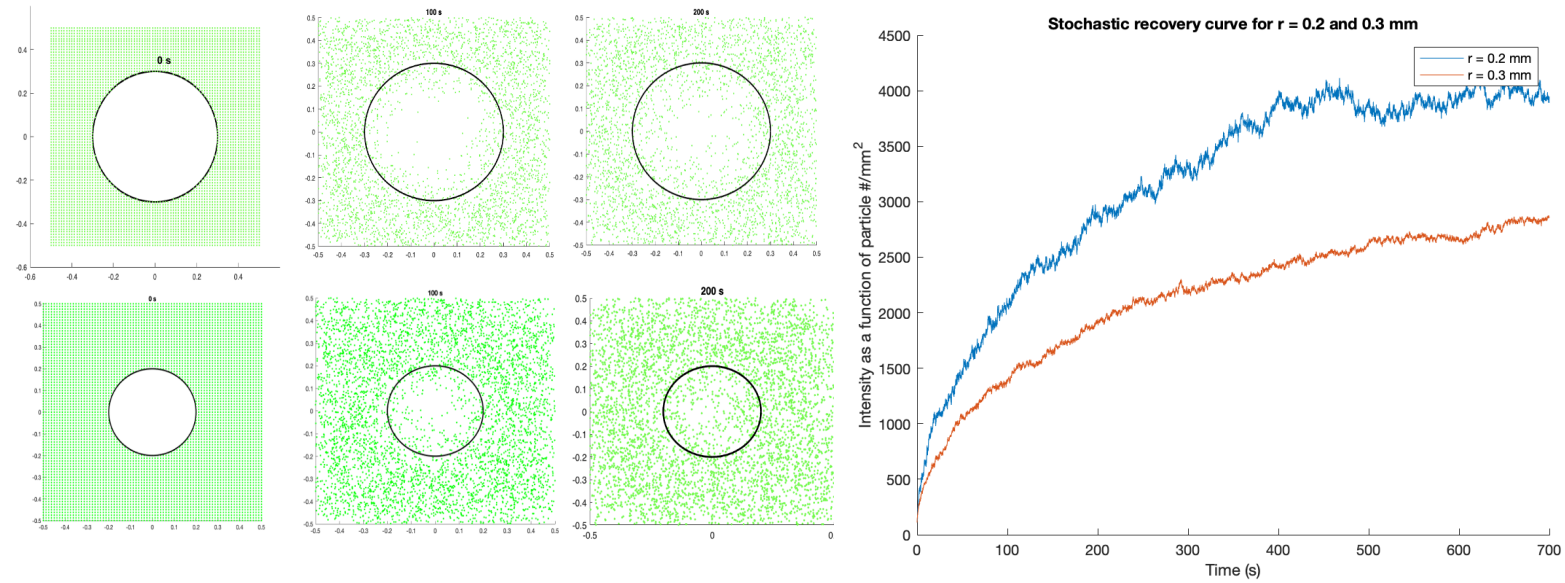


Figure 9: *Left: stochastic simulation of r = 0.3mm (top) and r = 0.2mm (bottom); right: recovery curves for both scenarios. Notice that the fluorescence recovers considerably faster when a circle of smaller radius is photobleached, which is both qualitatively apparent in the left figure and quantitatively apparent in the right. In particular, the recovery curve for 0.2mm both grows and plateaus faster than the recovery curve for 0.3mm; we have designed our simulation such that the total number of points remains constant, so greater intensity (i.e. particles/unit area in photobleached circle) implies faster recovery.*

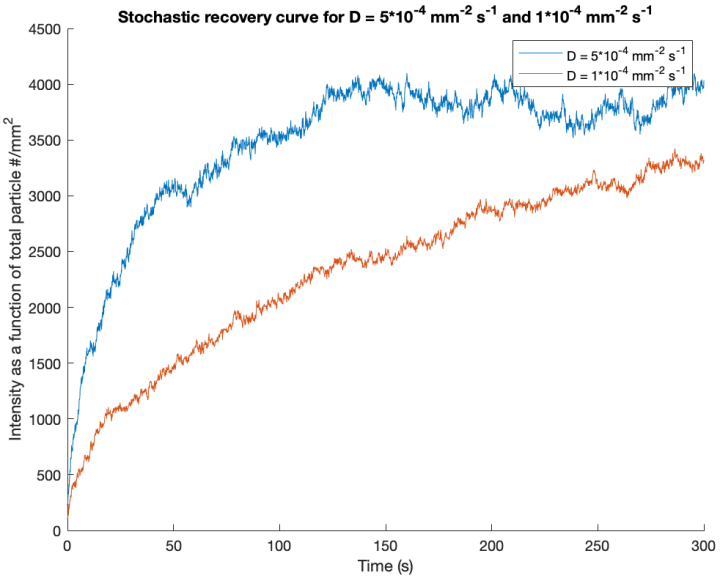


Figure 10: *stochastic recovery curves with changing D. As expected, a higher diffusion coefficient produces faster recovery (the particles will move more rapidly into the photobleached region).*

Conclusion and Outlook

Both deterministic and stochastic simulations for FRAP were successfully developed in MATLAB. Moreover, we have produced two models of cells (the size of which can be changed) by which we may model the photobleaching and consequent diffusion of a strict circular region (of which radius can be changed) assuming that the molecule of interest diffuses purely and according to diffusion coefficient D (which can also be changed). The model was tested for changes in diffusion coefficient and radius and behaved as expected in both: as such, we see that the model is both qualitatively and quantitatively quite close to experimental diffusion in FRAP. Furthermore, the development of a deterministic model and stochastic simulation permitted an analysis of the suitability of each model to FRAP; while we saw more noise in the models produced by the stochastic simulations, they prove more realistic to the circumstances by which molecules diffuse given the inclusion of a randomized Brownian noise factor.

However, the stochastic simulation suffers from a lack of efficiency, which is part of where future steps could be taken: since some loops are completed millions of times given the nature of checking the random walk, vectorizing these loops should produce a more efficient simulation that runs faster and produces less noisy results given more points. Furthermore, the current stochastic simulation assumes a distribution of massless and sizeless particles over continuous space, which is obviously not realistic to proteins in a cell. One can thus mimic a cell's environment more accurately by discretizing the particles and accounting for drift as well as possibly adding some form of selective barrier to account for binding.

Acknowledgments

Thank you to Mara Casebeer and Joonas Keski-Rahkonen for their invaluable help and input on the concepts and figures of the project; thank you also to Professor Adam Cohen, Lu Wang, and the rest of the teaching staff for making Chemistry 10 possible.

References

[1] Chudakov, D. M., Matz, M. V., Lukyanov, S., & Lukyanov, K. A. (2010). Fluorescent Proteins and Their Applications in Imaging Living Cells and Tissues. *Physiological Reviews*, *90*(3), 1103–1163. https://doi.org/10.1152/physrev.00038.2009

[2] Rapsomaniki, M. A., Cinquemani, E., Giakoumakis, N. N., Kotsantis, P., Lygeros, J., & Lygerou, Z. (2015). Inference of protein kinetics by stochastic modeling and simulation of fluorescence recovery after photobleaching experiments. *Bioinformatics*, *31*(3), 355–362. https://doi.org/10.1093/bioinformatics/btu619

[3] Rayan, G., Guet, J.-E., Taulier, N., Pincet, F., & Urbach, W. (2010). Recent Applications of Fluorescence Recovery after Photobleaching (FRAP) to Membrane Bio-Macromolecules. *Sensors*, *10*(6), 5927–5948. https://doi.org/10.3390/s100605927

[4] Sprague, B. L., Pego, R. L., Stavreva, D. A., & McNally, J. G. (2004). Analysis of Binding Reactions by Fluorescence Recovery after Photobleaching. *Biophysical Journal*, *86*(6), 3473–3495. https://doi.org/10.1529/biophysj.103.026765

[5] Deschout, H., Raemdonck, K., Demeester, J., De Smedt, S. C., & Braeckmans, K. (2014). FRAP in Pharmaceutical Research: Practical Guidelines and Applications in Drug Delivery. *Pharmaceutical Research*, *31*(2), 255–270. https://doi.org/10.1007/s11095-013-1146-9

[6] Erban, R., Chapman, J., & Maini, P. (2007). *A practical guide to stochastic simulations of reaction-diffusion processes*. https://doi.org/10.48550/ARXIV.0704.1908

[7] Hill, Christian. (n.d.). The 2D diffusion equation. In *Learning Scientific Programming with Python* (1st ed.). Cambridge University Press.

[8] Bläßle, A., Soh, G., Braun, T., Mörsdorf, D., Preiß, H., Jordan, B. M., & Müller, P. (2018). Quantitative diffusion measurements using the open-source software PyFRAP. *Nature Communications*, *9*(1), 1582. https://doi.org/10.1038/s41467-018-03975-6

[9] Chavanis, P.-H. (2019). The Generalized Stochastic Smoluchowski Equation. *Entropy*, *21*(10), 1006. https://doi.org/10.3390/e21101006

Final Project MATLAB Code

Photobleaching Process

Random walk for A decay (photobleaching process)

```
% stochastic simulation of simple reaction-diffusion decay
As = zeros(1, 5000);
As(1) = 20;
delt = 0.005; % sec
ts = zeros(1, 5000);
k = 0.1 % sec^-1

% array of times
for i = 1:5000
    ts(i) = 0 + delt*(i-1);
end

% loop thru 2 walks
for i = 1:2
    % loop steps per walk
    for j = 1:4999
        r = rand;
        if r < As(j)*k*delt
            As(j+1) = As(j)-1;
        else
            As(j+1) = As(j);
        end
    end
    figure(1)
    hold on
    plot(ts, As)
end
plot(ts, 20*exp(-k*ts), 'k')
xlabel('Time (s)')
ylabel('Number of particles')
title('Stochastic simulation of number of particles as function of time')
hold off

% more accurate stochastic simulation

k = 0.1 % sec^-1

% 10 realizations of walk
for i = 1:10
    temper = 1;
    As = zeros(1, temper);
    As(1) = 20; % initial A
    ts = zeros(1, temper);
```

```
    % keep the walk going until we run out of A
    while As(temper) ~= 0
        r = rand;
        t = ts(temper);
        tau = 1/(As(temper)*k) * log(1/r);
        ts(temper+1) = t+tau;
        As(temper+1) = As(temper) - 1;
        temper = temper+1;
    end
    hold on
    figure(2)
    stairs(ts, As)
end
xlabel('Time (s)')
ylabel('Number of particles')
title('Improved stochastic simulation of number of particles')
hold off
```

Deterministic simulation and recovery curve

Akin to heat convection. Suppose probability density is normalized such that the probability density of an empty region is 0 and the probability density of a full region is 400.

```
% cell size (mm)
w = 1;
h = 1;

% intervals in x- and y-directions, mm
dx = 0.01;
dy = 0.01;

% diffusivity of protein
D = 1e-4;
Tcool = 0; % empty
Thot = 400; % full
nx = w/dx;
ny = h/dy;
dx2 = dx*dx;
dy2 = dy*dy;
dt = dx2 * dy2 / (2 * D * (dx2 + dy2)); % maximum time interval before
destabilization
u0 = Tcool * ones(nx, ny); % set state to all empty
u = u0;

% Initial conditions - circle of radius r centred at (cx,cy) (mm)
r = 0.2;
cx = 0.5;
cy = 0.5;
r2 = r^2;
```

```
% For recovery curve: find maximal density of photobleached region
maxdense = 0;
for i = 1:nx
    for j = 1:ny
        p2 = (i*dx-cx)^2 + (j*dy-cy)^2; % distance to center
        if p2 < r2 % if further than radius (i.e. outside circle): max
probability density
            u0(i,j) = Thot;
            maxdense = maxdense+u0(i, j);
        end
    end
end

for i = 1:nx
    for j = 1:ny
        p2 = (i*dx-cx)^2 + (j*dy-cy)^2; % distance to center
        if p2 > r2 % if further than radius (i.e. outside circle): max
probability density
            u0(i,j) = Thot;
        else % reset everything else to empty again
            u0(i,j) = Tcool;
        end
    end
end

nsteps = 1201; % can be adjusted depending on what time interval

times = zeros(1, nsteps)
for i = 1:nsteps
    times(i) = 0 + (i-1)*dt*1000;
end

mfig = [1, 201, 401, 1201]; % can be adjusted: this is set up for 0, 50,
100, 300 secs given D = 1e-4
fignum = 0;
fig = figure;
averages = zeros(1, nsteps)
for m = 1:nsteps
    % for recovery curve: averages is the sum of probability density inside
    % photobleached circle, totals is the sum of all probability densities
    for i = 1:nx
        for j = 1:ny
            p2 = (i*dx-cx)^2 + (j*dy-cy)^2;
            if p2 < r2 % in circle: averages
                averages(m) = averages(m) + u0(i, j);
            end
        end
    end
    [u0, u] = do_timestep(u0, u); % code for function is at end
    % set boundary points to hot to prevent absorbance of particles
```

```
    for i = 1:100
        u(1, i) = Thot;
    end
    for j = 1:100
        u(100, j) = Thot;
    end
    for i = 1:100
        u(i, 1) = Thot;
    end
    for j = 1:100
        u(j, 100) = Thot;
    end
    if ismember(m, mfig) % if we specified time, plot state
        fignum = fignum + 1;
        disp([m, fignum]);
        ax = subplot(2, 2, fignum);
        im = imagesc(u, 'CDataMapping', 'scaled');
        colormap summer;
        clim([Tcool, Thot]);
        axis off;
        title(sprintf('%.1f ms', (m-1)*dt*1000));
    end
end
set(gcf, 'Position', [100, 100, 800, 800]);
cbar_ax = axes('Position', [0.9, 0.15, 0.03, 0.7]);
xlabel(cbar_ax, '$T$ / K', 'Interpreter', 'latex', 'FontSize', 12);
colorbar(cbar_ax);

% plot recovery curve
figure(3)
plot(times, averages./maxdense)
xlabel("Time elapsed (ms)")
ylabel("Intensity as a function of total # of particles")
title("Deterministic recovery curve")
```

Grid of points outside photobleaching circle

```
% circular random walk
a = 0:0.01:2*pi;  % angle
r = 0.2;           % radius
xv = r*cos(a);    % cartesian x coordinate
yv = r*sin(a);    % cartesian y coordinate
xq = 0.0125*repmat(-40:40,81,1);
yq = xq';
[in, on] = inpolygon(xq, yq, xv, yv); % boolean array of points in or on
the bleached circle

figure(4)
clear figure
hold on
```

```
axis([-0.5 0.5 -0.5 0.5]);
plot(xv,yv,'k','LineWidth', 2) % circle

% array of x and y outside circle
xys = zeros(2, 4772) % each column corresponds to a point
column = 1;
for x = 1:81
    for y = 1:81
        if in(x, y) == 0 && on(x, y) == 0 % plot if not in circle
            plot(0.0125*(x-41), 0.0125*(y-41), 'g.')
            xys(1, column) = 0.0125*(x-41);
            xys(2, column) = 0.0125*(y-41);
            column = column+1;
        end
    end
end
hold off
```

Random walking every point outside circle

```
D = 1e-4 % mm^2 sec^-1
Xs = zeros(5767, 7000); % storage array for all X-coordinates
Ys = zeros(5767, 7000); % storage array for all Y-coordinates
delt = 0.1;

for i = 1:5767 % note that 5767 is a temporary number that should be
adjusted if radius of circle/concentration of points is changed
    Xs(i, 1) = xys(1, i); % set initial point at t=0 to point not in circle
    Ys(i, 1) = xys(2, i);
    for k = 1:6999
        epsx = normrnd(0, 1);
        epsy = normrnd(0, 1);
        Xs(i, k+1) = Xs(i, k) + sqrt(2*D*delt)*epsx;
        Ys(i, k+1) = Ys(i, k) + sqrt(2*D*delt)*epsy;
        if abs(Xs(i, k+1)) > 0.5
            Xs(i, k+1) = Xs(i, k) - sqrt(2*D*delt)*epsx;
        end
        if abs(Ys(i, k+1)) > 0.5
            Ys(i, k+1) = Ys(i, k) - sqrt(2*D*delt)*epsy;
        end
    end
end
```

Plot all points at desired times (t = 50, 100, and 300 shown below).

```
figure(5)
hold on
axis([-0.5 0.5 -0.5 0.5]);
for i = 1:5767
```

```
    plot(Xs(i, 501), Ys(i, 501), 'g.'); % plot each point at t = 50
    plot(xv,yv,'k','LineWidth', 2)
    title('50 s')
end
hold off

figure(6)
hold on
axis([-0.5 0.5 -0.5 0.5]);
for i = 1:5767
    plot(Xs(i, 1001), Ys(i, 1001), 'g.'); % plot each point at t = 100
    plot(xv,yv,'k','LineWidth', 2)
    title('100 s')
end
hold off

figure(7)
hold on
axis([-0.5 0.5 -0.5 0.5]);
for i = 1:5767
    plot(Xs(i, 3001), Ys(i, 3001), 'g.'); % plot each point at t = 300
    plot(xv,yv,'k','LineWidth', 2)
    title('300 s')
end
hold off
```

Stochastic recovery curve

```
inscount = zeros(1, 6999) % count number of particles inside circle
ts = zeros(1, 6999) % time array
for i = 1:6999
    ts(i) = 0 + delt*(i-1);
end

for i = 1:5767
    for k = 1:6999
        xq = Xs(i, k);
        yq = Ys(i, k);
        [in, on] = inpolygon(xq, yq, xv, yv);
         if in == 1 || on == 1
            inscount(k) = inscount(k) + 1;
         end
    end
end

plot(ts, inscount./0.18)
xlabel('Time (s)')
ylabel('Intensity as a function of number of particles/mm^{2}')
title('Stochastic recovery curve')
```

Function as used in deterministic simulation

Derivation via finite differences method in Appendix 2.

```
function [u0, u] = do_timestep(u0, u)
    D = 2.5e-5;
    dx = 0.01;
    dy = 0.01;
    dx2 = dx*dx;
    dy2 = dy*dy;
    dt = dx2 * dy2 / (2 * D * (dx2 + dy2));
    u(2:end-1, 2:end-1) = u0(2:end-1, 2:end-1) + D * dt * ...
        ((u0(3:end, 2:end-1) - 2*u0(2:end-1, 2:end-1) + u0(1:end-2,
2:end-1))/dx2 + ...
        (u0(2:end-1, 3:end) - 2*u0(2:end-1, 2:end-1) + u0(2:end-1,
1:end-2))/dy2);
    u0 = u;
end
```

Appendix 2 – Notes on the derivation of equations

1. Derivation of model in figure 3: see [6]
2. Discretization of *U* via finite-differences:

Focusing our simulation on a 1-by-1 unit square cell gives the boundary conditions

$$0 \leq x \leq 1, 0 \leq y \leq 1,$$

and we approximate *U* as a discretized function $u^{(n)}_{i,j}$ for $x = i\Delta x$, $y = i\Delta y$, and $t = n\Delta i$. Applying finite-difference approximation as in [5] gives:

$$\frac{u_{i,j}^{(n+1)} - u_{i,j}^{(n)}}{\Delta t} = D\left[\frac{u_{i+1,j}^{(n)} - 2u_{i,j}^{(n)} + u_{i-1,j}^{(n)}}{(\Delta x)^2} + \frac{u_{i,j+1}^{(n)} - 2u_{i,j}^{(n)} + u_{i,j-1}^{(n)}}{(\Delta y)^2}\right],$$

and rearranging gives

$$u_{i,j}^{(n+1)} = u_{i,j}^{(n)} + D\Delta t\left[\frac{u_{i+1,j}^{(n)} - 2u_{i,j}^{(n)} + u_{i-1,j}^{(n)}}{(\Delta x)^2} + \frac{u_{i,j+1}^{(n)} - 2u_{i,j}^{(n)} + u_{i,j-1}^{(n)}}{(\Delta y)^2}\right],$$

from which we can obtain the probability density function of the system at time-step $n + 1$. See [7] for details.