CmpE 49G - Project 3 - Codes
Emre Girgin - 2016400099

1. With Enzymes

a. Board_sim_run.m

```
%% Set Parameters 1

sim_params1.rx_center               = [0, 0, 0];
sim_params1.rx_r_inMicroMeters      = 5;
sim_params1.rx_tx_distance          = 5;
sim_params1.tx_emission_pt          = sim_params1.rx_center +
[sim_params1.rx_tx_distance+sim_params1.rx_r_inMicroMeters, 0, 0];
sim_params1.D_inMicroMeterSqrPerSecond = 75;
sim_params1.lambda_degRate          = 5.4152;

sim_params1.tend                    = 0.4;
sim_params1.delta_t                 = 0.0001;
sim_params1.num_molecules           = 50000;

%% Set Parameters 2

sim_params2.rx_center               = [0, 0, 0];
sim_params2.rx_r_inMicroMeters      = 5;
sim_params2.rx_tx_distance          = 5;
sim_params2.tx_emission_pt          = sim_params2.rx_center +
[sim_params2.rx_tx_distance+sim_params2.rx_r_inMicroMeters, 0, 0];
sim_params2.D_inMicroMeterSqrPerSecond = 200;
sim_params2.lambda_degRate          = 5.4152;

sim_params2.tend                    = 0.4;
sim_params2.delta_t                 = 0.0001;
sim_params2.num_molecules           = 50000;

%% Set Parameters 3

sim_params3.rx_center               = [0, 0, 0];
sim_params3.rx_r_inMicroMeters      = 5;
sim_params3.rx_tx_distance          = 5;
sim_params3.tx_emission_pt          = sim_params3.rx_center +
[sim_params3.rx_tx_distance+sim_params3.rx_r_inMicroMeters, 0, 0];
sim_params3.D_inMicroMeterSqrPerSecond = 75;
sim_params3.lambda_degRate          = 10.8304;

sim_params3.tend                    = 0.4;
sim_params3.delta_t                 = 0.0001;
```

```
sim_params3.num_molecules           = 50000;

%% Set Parameters 4

sim_params4.rx_center               = [0, 0, 0];
sim_params4.rx_r_inMicroMeters      = 5;
sim_params4.rx_tx_distance          = 5;
sim_params4.tx_emission_pt          = sim_params4.rx_center +
[sim_params4.rx_tx_distance+sim_params4.rx_r_inMicroMeters, 0, 0];
sim_params4.D_inMicroMeterSqrPerSecond = 200;
sim_params4.lambda_degRate          = 10.8304;

sim_params4.tend                    = 0.4;
sim_params4.delta_t                 = 0.0001;
sim_params4.num_molecules           = 50000;

%% SIMULATE Set 1

fprintf('\nSimulation <sim_gaussianRW_Point2Spherical_FFP_3D> \t\t[START]')
tstart = tic;
[nrx_sim_timeline1, time1] = sim_gaussianRW_Point2Spherical_FFP_3D(sim_params1);
fprintf('\nSimulation <sim_gaussianRW_Point2Spherical_FFP_3D> \t\t[End] \tDuration = %f\n',
toc(tstart))

%% SIMULATE Set 2

fprintf('\nSimulation <sim_gaussianRW_Point2Spherical_FFP_3D> \t\t[START]')
tstart = tic;
[nrx_sim_timeline2, time2] = sim_gaussianRW_Point2Spherical_FFP_3D(sim_params2);
fprintf('\nSimulation <sim_gaussianRW_Point2Spherical_FFP_3D> \t\t[End] \tDuration = %f\n',
toc(tstart))
%% SIMULATE Set 3

fprintf('\nSimulation <sim_gaussianRW_Point2Spherical_FFP_3D> \t\t[START]')
tstart = tic;
[nrx_sim_timeline3, time3] = sim_gaussianRW_Point2Spherical_FFP_3D(sim_params3);
fprintf('\nSimulation <sim_gaussianRW_Point2Spherical_FFP_3D> \t\t[End] \tDuration = %f\n',
toc(tstart))

%% SIMULATE Set 4

fprintf('\nSimulation <sim_gaussianRW_Point2Spherical_FFP_3D> \t\t[START]')
tstart = tic;
[nrx_sim_timeline4, time4] = sim_gaussianRW_Point2Spherical_FFP_3D(sim_params4);
```

```
fprintf('\nSimulation <sim_gaussianRW_Point2Spherical_FFP_3D> \t\t[End] \tDuration = %f\n',
toc(tstart))



%% THEORETICAL NRX Set 1

fprintf('\nTheoretical Formula \t\t[START]')
tstart = tic;
[nrx_theory_timeline1] = eval_theoretical_nrx_3d_Point2Spherical_FFP_3D(sim_params1,
time1);
fprintf('\nTheoretical Formula  \t\t[End] \tDuration = %f\n', toc(tstart))

%% THEORETICAL NRX Set 2

fprintf('\nTheoretical Formula \t\t[START]')
tstart = tic;
[nrx_theory_timeline2] = eval_theoretical_nrx_3d_Point2Spherical_FFP_3D(sim_params2,
time2);
fprintf('\nTheoretical Formula  \t\t[End] \tDuration = %f\n', toc(tstart))

%% THEORETICAL NRX Set 3

fprintf('\nTheoretical Formula \t\t[START]')
tstart = tic;
[nrx_theory_timeline3] = eval_theoretical_nrx_3d_Point2Spherical_FFP_3D(sim_params3,
time3);
fprintf('\nTheoretical Formula  \t\t[End] \tDuration = %f\n', toc(tstart))

%% THEORETICAL NRX Set 4

fprintf('\nTheoretical Formula \t\t[START]')
tstart = tic;
[nrx_theory_timeline4] = eval_theoretical_nrx_3d_Point2Spherical_FFP_3D(sim_params4,
time4);
fprintf('\nTheoretical Formula  \t\t[End] \tDuration = %f\n', toc(tstart))



%% PLOT Set 1

merge_cnt = 10;
[nrx_sim_timeline_merged1, time_merged1] = helper_merge_timeline(merge_cnt,
nrx_sim_timeline1, time1);
```

```
[nrx_theory_timeline_merged1, ~] = helper_merge_timeline(merge_cnt, nrx_theory_timeline1,
time1);


hFig = figure;
set(gcf,'PaperPositionMode','auto')
set(hFig, 'Position', [0 101 600 400])

subplot(2,1,1)
plot(time1, cumsum(nrx_sim_timeline1)/sim_params1.num_molecules, '-', 'LineWidth', 2)
hold on
plot(time1, cumsum(nrx_theory_timeline1), '--', 'LineWidth', 2)
grid on
xlabel('Time - (s)')
ylabel('The Number of Molecules Hitting Receiver Before Decomposition')
legend('Param Set 1', 'Theory');
title(['\Deltat=', num2str(sim_params1.delta_t), '; r_{rx}=',
num2str(sim_params1.rx_r_inMicroMeters), '; dist=', num2str(sim_params1.rx_tx_distance), ';
D=', num2str(sim_params1.D_inMicroMeterSqrPerSecond), '; \lambda=',
num2str(sim_params1.lambda_degRate) ])
hold off

subplot(2,1,2)
plot(time_merged1, nrx_sim_timeline_merged1/sim_params1.num_molecules, '-', 'LineWidth', 2)
hold on
plot(time_merged1, nrx_theory_timeline_merged1, '--', 'LineWidth', 2)
grid on
xlabel('Time - (s)')
ylabel('Average Fraction of Received Molecules in \Delta t')
legend('Param Set 1', 'Theory');
title(['\Deltat=', num2str(sim_params1.delta_t), '; r_{rx}=',
num2str(sim_params1.rx_r_inMicroMeters), '; dist=', num2str(sim_params1.rx_tx_distance), ';
D=', num2str(sim_params1.D_inMicroMeterSqrPerSecond), '; \lambda=',
num2str(sim_params1.lambda_degRate) ])
hold off

%% PLOT Set 2

merge_cnt = 10;
[nrx_sim_timeline_merged2, time_merged2] = helper_merge_timeline(merge_cnt,
nrx_sim_timeline2, time2);
[nrx_theory_timeline_merged2, ~] = helper_merge_timeline(merge_cnt, nrx_theory_timeline2,
time2);
```

```
hFig = figure;
set(gcf,'PaperPositionMode','auto')
set(hFig, 'Position', [0 101 600 400])

subplot(2,1,1)
plot(time2, cumsum(nrx_sim_timeline2)/sim_params2.num_molecules, '-', 'LineWidth', 2)
hold on
plot(time2, cumsum(nrx_theory_timeline2), '--', 'LineWidth', 2)
grid on
xlabel('Time - (s)')
ylabel('The Number of Molecules Hitting Receiver Before Decomposition')
legend('Param Set 2', 'Theory');
title(['\Deltat=', num2str(sim_params2.delta_t), '; r_{rx}=',
num2str(sim_params2.rx_r_inMicroMeters), '; dist=', num2str(sim_params2.rx_tx_distance), ';
D=', num2str(sim_params2.D_inMicroMeterSqrPerSecond), '; \lambda=',
num2str(sim_params2.lambda_degRate)])
hold off

subplot(2,1,2)
plot(time_merged2, nrx_sim_timeline_merged2/sim_params2.num_molecules, '-', 'LineWidth', 2)
hold on
plot(time_merged2, nrx_theory_timeline_merged2, '--', 'LineWidth', 2)
grid on
xlabel('Time - (s)')
ylabel('Average Fraction of Received Molecules in \Delta t')
legend('Param Set 2', 'Theory');
title(['\Deltat=', num2str(sim_params2.delta_t), '; r_{rx}=',
num2str(sim_params2.rx_r_inMicroMeters), '; dist=', num2str(sim_params2.rx_tx_distance), ';
D=', num2str(sim_params2.D_inMicroMeterSqrPerSecond), '; \lambda=',
num2str(sim_params2.lambda_degRate)])
hold off

%% PLOT Set 3

merge_cnt = 10;
[nrx_sim_timeline_merged3, time_merged3] = helper_merge_timeline(merge_cnt,
nrx_sim_timeline3, time3);
[nrx_theory_timeline_merged3, ~] = helper_merge_timeline(merge_cnt, nrx_theory_timeline3,
time3);

hFig = figure;
set(gcf,'PaperPositionMode','auto')
set(hFig, 'Position', [0 101 600 400])
```

```
subplot(2,1,1)
plot(time3, cumsum(nrx_sim_timeline3)/sim_params3.num_molecules, '-', 'LineWidth', 2)
hold on
plot(time3, cumsum(nrx_theory_timeline3), '--', 'LineWidth', 2)
grid on
xlabel('Time - (s)')
ylabel('The Number of Molecules Hitting Receiver Before Decomposition')
legend('Param Set 3', 'Theory');
title(['\Deltat=', num2str(sim_params3.delta_t), '; r_{rx}=',
num2str(sim_params3.rx_r_inMicroMeters), '; dist=', num2str(sim_params3.rx_tx_distance), ';
D=', num2str(sim_params3.D_inMicroMeterSqrPerSecond), '; \lambda=',
num2str(sim_params3.lambda_degRate)])
hold off

subplot(2,1,2)
plot(time_merged3, nrx_sim_timeline_merged3/sim_params3.num_molecules, '-', 'LineWidth', 2)
hold on
plot(time_merged3, nrx_theory_timeline_merged3, '--', 'LineWidth', 2)
grid on
xlabel('Time - (s)')
ylabel('Average Fraction of Received Molecules in \Delta t')
legend('Param Set 3', 'Theory');
title(['\Deltat=', num2str(sim_params3.delta_t), '; r_{rx}=',
num2str(sim_params3.rx_r_inMicroMeters), '; dist=', num2str(sim_params3.rx_tx_distance), ';
D=', num2str(sim_params3.D_inMicroMeterSqrPerSecond), '; \lambda=',
num2str(sim_params3.lambda_degRate)])
hold off


%% PLOT Set 4

merge_cnt = 10;
[nrx_sim_timeline_merged4, time_merged4] = helper_merge_timeline(merge_cnt,
nrx_sim_timeline4, time4);
[nrx_theory_timeline_merged4, ~] = helper_merge_timeline(merge_cnt, nrx_theory_timeline4,
time4);

hFig = figure;
set(gcf,'PaperPositionMode','auto')
set(hFig, 'Position', [0 101 600 400])

subplot(2,1,1)
plot(time4, cumsum(nrx_sim_timeline4)/sim_params4.num_molecules, '-', 'LineWidth', 2)
hold on
```

```
plot(time4, cumsum(nrx_theory_timeline4), '--', 'LineWidth', 2)
grid on
xlabel('Time - (s)')
ylabel('The Number of Molecules Hitting Receiver Before Decomposition')
legend('Param Set 4', 'Theory');
title(['\Deltat=', num2str(sim_params4.delta_t), '; r_{rx}=',
num2str(sim_params4.rx_r_inMicroMeters), '; dist=', num2str(sim_params4.rx_tx_distance), ';
D=', num2str(sim_params4.D_inMicroMeterSqrPerSecond), '; \lambda=',
num2str(sim_params4.lambda_degRate)])
hold off

subplot(2,1,2)
plot(time_merged4, nrx_sim_timeline_merged4/sim_params4.num_molecules, '-', 'LineWidth', 2)
hold on
plot(time_merged4, nrx_theory_timeline_merged4, '--', 'LineWidth', 2)
grid on
xlabel('Time - (s)')
ylabel('Average Fraction of Received Molecules in \Delta t')
legend('Param Set 4', 'Theory');
title(['\Deltat=', num2str(sim_params4.delta_t), '; r_{rx}=',
num2str(sim_params4.rx_r_inMicroMeters), '; dist=', num2str(sim_params4.rx_tx_distance), ';
D=', num2str(sim_params4.D_inMicroMeterSqrPerSecond), '; \lambda=',
num2str(sim_params4.lambda_degRate)])
hold off
```

b. eval_theoretical_nrx_3d_Point2Spherical_FFP_3D.m

```
function [ frx_t ] = eval_theoretical_nrx_3d_Point2Spherical_FFP_3D( sim_params, time )

dist        = sim_params.rx_tx_distance;
rx_r        = sim_params.rx_r_inMicroMeters;
D           = sim_params.D_inMicroMeterSqrPerSecond;

part1 = rx_r / (dist + rx_r);
part2_sub1 = exp(-dist*sqrt(sim_params.lambda_degRate/D)) * erfc(dist./sqrt(4*D*time) -
sqrt(sim_params.lambda_degRate*time));
part2_sub2 = exp(dist*sqrt(sim_params.lambda_degRate/D)) * erfc(dist./sqrt(4*D*time) +
sqrt(sim_params.lambda_degRate*time));
nrx_cumulative = (1/2) * part1 * (part2_sub1 + part2_sub2);

% If you subtract 1-shifted version from it self, you end up with nrx at each simulation step
frx_t = nrx_cumulative - [0 nrx_cumulative(1:end-1)];
```

```
end
```

c. sim_gaussianRW_Point2Spherical_FFP_3D.m

```
function [ nRx_timeline, time ] = sim_gaussianRW_Point2Spherical_FFP_3D( sim_params )

rx_center                  = sim_params.rx_center;
rx_r_inMicroMeters         = sim_params.rx_r_inMicroMeters;

tx_emission_pt             = sim_params.tx_emission_pt;
D                          = sim_params.D_inMicroMeterSqrPerSecond;

tend               = sim_params.tend;
delta_t            = sim_params.delta_t;
num_molecules      = sim_params.num_molecules;

% Standard deviation of step size of movement N(0,sigma)
sigma = (2*D*delta_t)^0.5;

% Square of the Rx Radius is useful for checking the hit action, it doesn't change so evaluating
once is enough
rx_membrane_sq = rx_r_inMicroMeters^2;


sim_step_count = round(tend/delta_t);
nRx_timeline = zeros (1, sim_step_count); % we are using only one-type of molecule

% Create molecules with INITIAL Coords: replicate num_molecules times
mol_coords_BEFORE_movement = repmat(tx_emission_pt, num_molecules, 1);

for ii = 1:sim_step_count
   mol_displacement = normrnd(0, sigma, size(mol_coords_BEFORE_movement,1), 3);
   mol_cords_AFTER_movement = mol_coords_BEFORE_movement + mol_displacement;

   % Generate random numbers to
   mol_degrade_prob = rand(size(mol_cords_AFTER_movement,1), 1);

   % Create a mask for non-degrading molecules
   nondegrade_mask = mol_degrade_prob < exp(-sim_params.lambda_degRate*delta_t);

   %disp(size(mol_cords_AFTER_movement))
```

```
    % Get only the molecules that do not degrade
    mol_cords_AFTER_movement = mol_cords_AFTER_movement(nondegrade_mask, :);
    %disp(size(mol_cords_AFTER_movement))

    %calculates the square of the distances between molecules to the Rx Center
    dist_sq_2_rcv_center = sum(bsxfun(@minus, mol_cords_AFTER_movement, rx_center).^2,
2);

    %checks if the molecules are outside of the receiver (NOT HIT)
    outside_rx_membrane_mask = dist_sq_2_rcv_center > rx_membrane_sq;

    %calculate the hit of molecules to the receiver (FOR THIS SIM STEP, NOT TOTAL)
    nRx_timeline(ii) = nRx_timeline(ii) + nnz(~outside_rx_membrane_mask );

    % Just keep the ones that did NOT hit, and do same things again
    mol_coords_BEFORE_movement =
mol_cords_AFTER_movement(outside_rx_membrane_mask, :);
end

time = delta_t:delta_t:tend;

end
```

d. Helper_merge_timeline.m

```
function [nrx_timeline_merged, time_merged] = helper_merge_timeline(merge_cnt,
nrx_timeline, time)

size_nrx_timeline = size(nrx_timeline, 2);

new_t_size = round(size_nrx_timeline/merge_cnt);

nrx_timeline_merged = sum( reshape(nrx_timeline, [merge_cnt, new_t_size]) );

time_merged = reshape(time, [merge_cnt, new_t_size]);
time_merged = time_merged(1,:);

end
```

2. Without Enzymes

a. Board_sim_run.m

```
%% Set Parameters 1

sim_params1.rx_center               = [0, 0, 0];
sim_params1.rx_r_inMicroMeters      = 5;
sim_params1.rx_tx_distance          = 5;
sim_params1.tx_emission_pt          = sim_params1.rx_center +
[sim_params1.rx_tx_distance+sim_params1.rx_r_inMicroMeters, 0, 0];
sim_params1.D_inMicroMeterSqrPerSecond = 75;

sim_params1.tend                    = 0.4;
sim_params1.delta_t                 = 0.0001;
sim_params1.num_molecules           = 50000;

%% Set Parameters 2

sim_params2.rx_center               = [0, 0, 0];
sim_params2.rx_r_inMicroMeters      = 5;
sim_params2.rx_tx_distance          = 5;
sim_params2.tx_emission_pt          = sim_params2.rx_center +
[sim_params2.rx_tx_distance+sim_params2.rx_r_inMicroMeters, 0, 0];
sim_params2.D_inMicroMeterSqrPerSecond = 200;

sim_params2.tend                    = 0.4;
sim_params2.delta_t                 = 0.0001;
sim_params2.num_molecules           = 50000;

%% SIMULATE Set 1

fprintf('\nSimulation <sim_gaussianRW_Point2Spherical_FFP_3D> \t\t[START]')
tstart = tic;
[nrx_sim_timeline1, time1] = sim_gaussianRW_Point2Spherical_FFP_3D(sim_params1);
fprintf('\nSimulation <sim_gaussianRW_Point2Spherical_FFP_3D> \t\t[End] \tDuration = %f\n',
toc(tstart))

%% SIMULATE Set 2

fprintf('\nSimulation <sim_gaussianRW_Point2Spherical_FFP_3D> \t\t[START]')
tstart = tic;
[nrx_sim_timeline2, time2] = sim_gaussianRW_Point2Spherical_FFP_3D(sim_params2);
```

```
fprintf('\nSimulation <sim_gaussianRW_Point2Spherical_FFP_3D> \t\t[End] \tDuration = %f\n',
toc(tstart))


%% THEORETICAL NRX Set 1

fprintf('\nTheoretical Formula \t\t[START]')
tstart = tic;
[nrx_theory_timeline1] = eval_theoretical_nrx_3d_Point2Spherical_FFP_3D(sim_params1,
time1);
fprintf('\nTheoretical Formula  \t\t[End] \tDuration = %f\n', toc(tstart))

%% THEORETICAL NRX Set 2

fprintf('\nTheoretical Formula \t\t[START]')
tstart = tic;
[nrx_theory_timeline2] = eval_theoretical_nrx_3d_Point2Spherical_FFP_3D(sim_params2,
time2);
fprintf('\nTheoretical Formula  \t\t[End] \tDuration = %f\n', toc(tstart))


%% PLOT Set 1
merge_cnt = 10;
[nrx_sim_timeline_merged1, time_merged1] = helper_merge_timeline(merge_cnt,
nrx_sim_timeline1, time1);
[nrx_theory_timeline_merged1, ~] = helper_merge_timeline(merge_cnt, nrx_theory_timeline1,
time1);

hFig = figure;
set(gcf,'PaperPositionMode','auto')
set(hFig, 'Position', [0 101 600 400])

subplot(2,1,1)
plot(time1, cumsum(nrx_sim_timeline1)/sim_params1.num_molecules, '-', 'LineWidth', 2)
hold on
plot(time1, cumsum(nrx_theory_timeline1), '--', 'LineWidth', 2)
grid on
xlabel('Time - (s)')
ylabel('Number of Molecules Hitting Receiver')
legend('Param Set 1', 'Theory');
title(['\Deltat=', num2str(sim_params1.delta_t), '; r_{rx}=',
num2str(sim_params1.rx_r_inMicroMeters), '; dist=', num2str(sim_params1.rx_tx_distance), ';
D=', num2str(sim_params1.D_inMicroMeterSqrPerSecond)])
hold off
```

```
subplot(2,1,2)
plot(time_merged1, nrx_sim_timeline_merged1/sim_params1.num_molecules, '-', 'LineWidth', 2)
hold on
plot(time_merged1, nrx_theory_timeline_merged1, '--', 'LineWidth', 2)
grid on
xlabel('Time - (s)')
ylabel('Average Fraction of Received Molecules in \Delta t')
legend('Param Set 1', 'Theory');
title(['\Deltat=', num2str(sim_params1.delta_t), '; r_{rx}=',
num2str(sim_params1.rx_r_inMicroMeters), '; dist=', num2str(sim_params1.rx_tx_distance), ';
D=', num2str(sim_params1.D_inMicroMeterSqrPerSecond)])
hold off

%% PLOT Set 2
merge_cnt = 10;
[nrx_sim_timeline_merged2, time_merged2] = helper_merge_timeline(merge_cnt,
nrx_sim_timeline2, time2);
[nrx_theory_timeline_merged2, ~] = helper_merge_timeline(merge_cnt, nrx_theory_timeline2,
time2);

hFig = figure;
set(gcf,'PaperPositionMode','auto')
set(hFig, 'Position', [0 101 600 400])

subplot(2,1,1)
plot(time2, cumsum(nrx_sim_timeline2)/sim_params2.num_molecules, '-', 'LineWidth', 2)
hold on
plot(time2, cumsum(nrx_theory_timeline2), '--', 'LineWidth', 2)
grid on
xlabel('Time - (s)')
ylabel('Number of Molecules Hitting Receiver')
legend('Param Set 2', 'Theory');
title(['\Deltat=', num2str(sim_params2.delta_t), '; r_{rx}=',
num2str(sim_params2.rx_r_inMicroMeters), '; dist=', num2str(sim_params2.rx_tx_distance), ';
D=', num2str(sim_params2.D_inMicroMeterSqrPerSecond)])
hold off

subplot(2,1,2)
plot(time_merged2, nrx_sim_timeline_merged2/sim_params2.num_molecules, '-', 'LineWidth', 2)
hold on
plot(time_merged2, nrx_theory_timeline_merged2, '--', 'LineWidth', 2)
grid on
xlabel('Time - (s)')
```

```
ylabel('Average Fraction of Received Molecules in \Delta t')
legend('Param Set 2', 'Theory');
title(['\Deltat=', num2str(sim_params2.delta_t), '; r_{rx}=',
num2str(sim_params2.rx_r_inMicroMeters), '; dist=', num2str(sim_params2.rx_tx_distance), ';
D=', num2str(sim_params2.D_inMicroMeterSqrPerSecond)])
hold off
```

b. eval_theoretical_nrx_3d_Point2Spherical_FFP_3D.m

```
function [ frx_t ] = eval_theoretical_nrx_3d_Point2Spherical_FFP_3D( sim_params, time )

dist          = sim_params.rx_tx_distance;
rx_r          = sim_params.rx_r_inMicroMeters;
D             = sim_params.D_inMicroMeterSqrPerSecond;

part1 = rx_r / (dist + rx_r);
nrx_cumulative = part1 * erfc(dist./sqrt(4*D*time));

% If you subtract 1-shifted version from it self, you end up with nrx at each simulation step
frx_t = nrx_cumulative - [0 nrx_cumulative(1:end-1)];

end
```

c. sim_gaussianRW_Point2Spherical_FFP_3D.m

```
function [ nRx_timeline, time ] = sim_gaussianRW_Point2Spherical_FFP_3D( sim_params )

rx_center                = sim_params.rx_center;
rx_r_inMicroMeters       = sim_params.rx_r_inMicroMeters;

tx_emission_pt           = sim_params.tx_emission_pt;
D                        = sim_params.D_inMicroMeterSqrPerSecond;

tend             = sim_params.tend;
delta_t          = sim_params.delta_t;
num_molecules    = sim_params.num_molecules;

% Standard deviation of step size of movement N(0,sigma)
```

```
sigma = (2*D*delta_t)^0.5;

% Square of the Rx Radius is useful for checking the hit action, it doesn't change so evaluating
once is enough
rx_membrane_sq = rx_r_inMicroMeters^2;


sim_step_count = round(tend/delta_t);
nRx_timeline = zeros (1, sim_step_count); % we are using only one-type of molecule

% Create molecules with INITIAL Coords: replicate num_molecules times
mol_coords_BEFORE_movement = repmat(tx_emission_pt, num_molecules, 1);

for ii = 1:sim_step_count
   mol_displacement = normrnd(0, sigma, size(mol_coords_BEFORE_movement,1), 3);
   mol_cords_AFTER_movement = mol_coords_BEFORE_movement + mol_displacement;

   %calculates the square of the distances between molecules to the Rx Center
   dist_sq_2_rcv_center = sum(bsxfun(@minus, mol_cords_AFTER_movement, rx_center).^2,
2);

   %checks if the molecules are outside of the receiver (NOT HIT)
   outside_rx_membrane_mask = dist_sq_2_rcv_center > rx_membrane_sq;

   %calculate the hit of molecules to the receiver (FOR THIS SIM STEP, NOT TOTAL)
   nRx_timeline(ii) = nRx_timeline(ii) + nnz(~outside_rx_membrane_mask );

   % Just keep the ones that did NOT hit, and do same things again
   mol_coords_BEFORE_movement =
mol_cords_AFTER_movement(outside_rx_membrane_mask, :);
end

time = delta_t:delta_t:tend;

end
```

d. helper_merge_timeline.m

```
function [nrx_timeline_merged, time_merged] = helper_merge_timeline(merge_cnt,
nrx_timeline, time)

size_nrx_timeline = size(nrx_timeline, 2);
```

```
new_t_size = round(size_nrx_timeline/merge_cnt);

nrx_timeline_merged = sum( reshape(nrx_timeline, [merge_cnt, new_t_size]) );

time_merged = reshape(time, [merge_cnt, new_t_size]);
time_merged = time_merged(1,:);

end
```