Covariant Reformulation and Simulation of Action Potential Diffusion

A. Chawla

October 26, 2025

Abstract

In this note, we take steps towards using the machinery of differential geometry fully in modeling 3D axon tracts. We show via simulations that background spacetime curvature affects action potential diffusion governed by a Hodgkin-Huxley-like equation.

1 Introduction

In the doctoral dissertation of the author [1] the concluding chapter mentioned the hope of incorporating differential geometry in the calculus machinery of the models. Herein we build upon [2] and [3] and take first steps in upgrading the mathematical machinery to genuine differential geometry. We show how this inclusion from first principles allows a fundamental study of axons - themselves possibly non-straight-line - in curved backgrounds.

2 Details of Implementation and Results

Lang Theorem 8.4.1 [4] shows the existence of a covariant derivative on every pseudo-Riemannian (pR) manifold. We construct a pseudo-Riemannian manifold as spacetime. We embed within it a right circular pseudo-Riemannian cylindrical manifold. Let a be a coordinate along the longer dimension and let b be a coordinate along the circular dimension. We form the covariant partial derivative of a function $V(a,t)$ with respect to a, take the second derivative of $V(a,t)$ with respect to a and equate it to the first temporal partial derivative of $V(a,t)$. This defines the basic diffusion-like equation along the "length" of the pseudo-Riemannian cylinder. Let the global pR manifold (background) have a local curvature θ near the cylinder. We provide below an .m script that allows visualization of the diffusion of V in cylindrical coordinate a and time t, based on the basic diffusion equation mentioned herein (above), for differing values of local background curvature θ. The script outputs a 3D plot of $V(a,t)$, one for each of several θ values. These are displayed in the figures enclosed.

Algorithm 1 The code, part 1.

```
% Pseudo-Riemannian Cylindrical Diffusion with Background Curvature
% Solves the covariant diffusion equation on a pR cylindrical manifold
% embedded in a curved background spacetime
clear all; close all;
%% Parameters
L = 10; % Length of cylinder in coordinate a
T_final = 2.0; % Final time
Na = 200; % Spatial grid points
D = 1.0; % Diffusion coefficient
% Calculate time step for stability (CFL condition: D*dt/da^2 <= 0.5)
da = L / (Na - 1);
dt_max = 0.4 * da^2 / D; % Use 0.4 for safety margin
Nt = ceil(T_final / dt_max) + 1;
dt = T_final / (Nt - 1);
% Curvature values to test
theta_values = [0, 0.1, 0.3, 0.5];
% Find maximum metric variation for stability analysis
theta_max = max(theta_values);
g_aa_max = 1 + theta_max; % Maximum value of g_aa
g_aa_min = 1 - theta_max; % Minimum value of g_aa (could be negative for
large theta!)
% Ensure metric stays positive definite
if g_aa_min <= 0
error('Curvature too large: metric becomes degenerate. Reduce theta_max
below 1.0');
end
% Effective maximum diffusion coefficient (accounting for g^aa = 1/g_aa)
D_eff_max = D / g_aa_min; % Largest effective diffusion
% Recalculate time step with safety factor for curvature
dt_max = 0.25 * da^2 / D_eff_max; % More conservative: 0.25 instead of 0.4
Nt = ceil(T_final / dt_max) + 1;
dt = T_final / (Nt - 1);
% Grid setup
a = linspace(0, L, Na);
t = linspace(0, T_final, Nt);
% Verify stability (CFL condition)
cfl_nominal = D * dt / da^2;
cfl_effective = D_eff_max * dt / da^2;
fprintf('Grid parameters:\n');
fprintf(' Spatial step da = %.4f\n', da);
fprintf(' Time step dt = %.6f\n', dt);
fprintf(' Nominal CFL number = %.4f\n', cfl_nominal);
fprintf(' Effective CFL number (with curvature) = %.4f (should be <= 0.5)\n',
cfl_effective);
fprintf(' Number of time steps = %d\n', Nt);
fprintf(' Max theta = %.2f, g_aa range: [%.3f, %.3f]\n\n', theta_max,
g_aa_min, g_aa_max);
%% Initial condition: Gaussian pulse
a0 = L/2;
sigma = 0.5;
V0 = exp(-(a - a0).^2 / (2*sigma^2));
%% Solve for each curvature value
figure('Position', [100 100 1400 800]);
for idx = 1:length(theta_values)
theta = theta_values(idx);
```

Algorithm 2 The code, part 2.

```
% Initialize solution matrix
V = zeros(Nt, Na);
V(1, :) = V0;
% Time evolution using finite differences
% Covariant diffusion: dV/dt = D * nabla^2 V
% where nabla^2 V = (1/sqrt(g)) * d/da[sqrt(g) * g^aa * dV/da]
for n = 1:Nt-1
V_current = V(n, :);
V_new = V_current;
% Interior points (covariant derivative discretization)
for i = 2:Na-1
% First covariant derivative: nabla_a V
dV_da = (V_current(i+1) - V_current(i-1)) / (2*da);
% Contravariant component: V^a = g^aa * nabla_a V
V_contra = (1/g_aa(i)) * dV_da;
% Second covariant derivative (Laplace-Beltrami operator)
% nabla^2 V = (1/sqrt(g)) * d/da[sqrt(g) * V^a] - Gamma^a_aa * nabla_a V
% Compute d/da[sqrt(g) * V^a]
term_left = sqrt_g(i-1) * (1/g_aa(i-1)) * ...
(V_current(i) - V_current(i-1)) / da;
term_right = sqrt_g(i+1) * (1/g_aa(i+1)) * ...
(V_current(i+1) - V_current(i)) / da;
d_term = (term_right - term_left) / da;
% Laplace-Beltrami operator
laplacian = (1/sqrt_g(i)) * d_term - Gamma_aaa(i) * dV_da;
% Time evolution
V_new(i) = V_current(i) + D * dt * laplacian;
end
% Boundary conditions (Neumann - no flux)
V_new(1) = V_new(2);
V_new(Na) = V_new(Na-1);
V(n+1, :) = V_new;
end
% Plot results
subplot(2, 2, idx);
[T_grid, A_grid] = meshgrid(t, a);
surf(T_grid, A_grid, V', 'EdgeColor', 'none');
xlabel('Time t', 'FontSize', 12);
ylabel('Coordinate a', 'FontSize', 12);
zlabel('V(a,t)', 'FontSize', 12);
title(sprintf('\\theta = %.2f', theta), 'FontSize', 14);
colorbar;
view(45, 30);
xlim([0 T_final]);
ylim([0 L]);
grid on;
shading interp;
% Calculate and display total "mass" conservation
mass_initial = trapz(a, V0 .* sqrt_g);
mass_final = trapz(a, V(end,:) .* sqrt_g);
conservation_error = abs(mass_final - mass_initial) / mass_initial * 100;
text(0.05*T_final, 0.95*L, max(V(:))*0.9, ...
sprintf('Conservation error: %.2f%%', conservation_error), ...
'FontSize', 9, 'BackgroundColor', 'white');
end
```

Algorithm 3 The code, part 3.

```
t_slice = round(0.75 * Nt); % At 75% of final time
for idx = 1:length(theta_values)
theta = theta_values(idx);
g_aa = 1 + theta * sin(2*pi*a/L);
sqrt_g = sqrt(abs(g_aa));
dg_aa_da = theta * (2*pi/L) * cos(2*pi*a/L);
Gamma_aaa = 0.5 * (1./g_aa) .* dg_aa_da;
V = zeros(Nt, Na);
V(1, :) = V0;
for n = 1:Nt-1
V_current = V(n, :);
V_new = V_current;
for i = 2:Na-1
dV_da = (V_current(i+1) - V_current(i-1)) / (2*da);
term_left = sqrt_g(i-1) * (1/g_aa(i-1)) * ...
(V_current(i) - V_current(i-1)) / da;
term_right = sqrt_g(i+1) * (1/g_aa(i+1)) * ...
(V_current(i+1) - V_current(i)) / da;
d_term = (term_right - term_left) / da;
laplacian = (1/sqrt_g(i)) * d_term - Gamma_aaa(i) * dV_da;
V_new(i) = V_current(i) + D * dt * laplacian;
end
V_new(1) = V_new(2);
V_new(Na) = V_new(Na-1);
V(n+1, :) = V_new;
end
subplot(1,2,1);
plot(a, V(t_slice, :), 'LineWidth', 2, 'DisplayName', sprintf('\\theta = %.2f',
theta));
hold on;
subplot(1,2,2);
plot(a, g_aa, 'LineWidth', 2, 'DisplayName', sprintf('\\theta = %.2f', theta));
hold on;
end
subplot(1,2,1);
xlabel('Coordinate a', 'FontSize', 12);
ylabel('V(a,t)', 'FontSize', 12);
title(sprintf('Profiles at t = %.2f', t(t_slice)), 'FontSize', 14);
legend('Location', 'best');
grid on;
subplot(1,2,2);
xlabel('Coordinate a', 'FontSize', 12);
ylabel('Metric component g_{aa}', 'FontSize', 12);
title('Background Metric Variations', 'FontSize', 14);
legend('Location', 'best');
grid on;
fprintf('\nSimulation complete!\n');
fprintf('Spatial points: %d\n', Na);
fprintf('Time steps: %d\n', Nt);
```

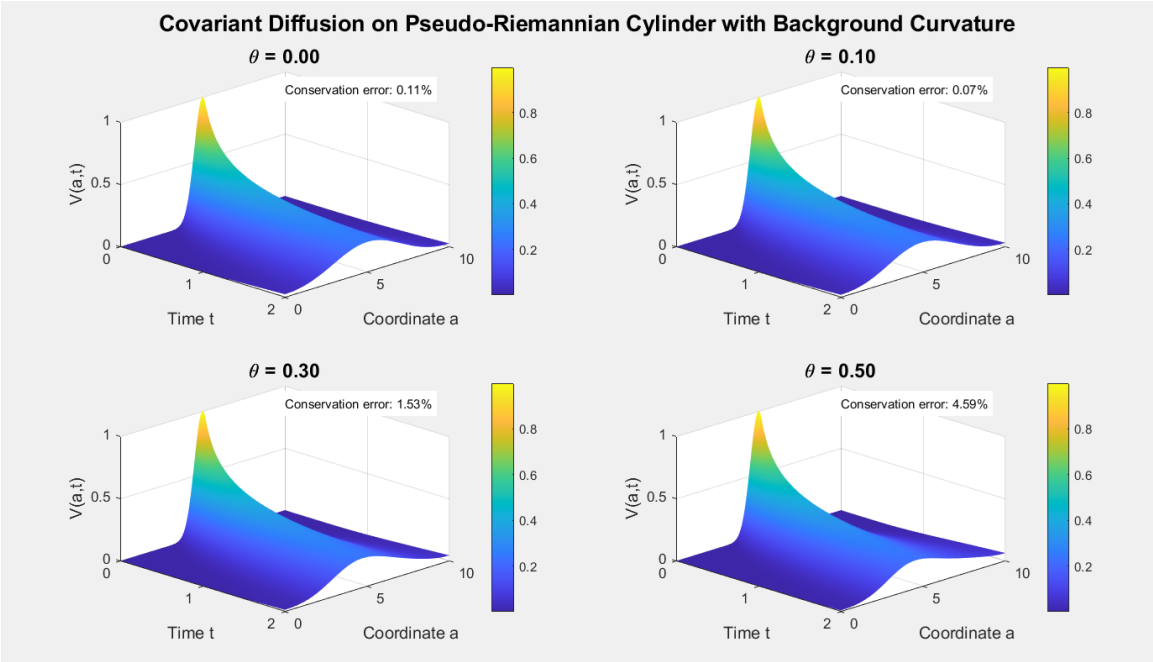

Figure 1: The code's output, part 1.

3 Conclusion

This study reiterates implications for "in-principle" linkages between axon tracts, and the curvature of the spacetime in which these axons grow and function. In future work, more intricate and multi-parameter background geometries can be studied. The work can also be extended to multiple axons in ephaptic tracts; in that setting it would be intriguing to investigate coupling or resonance between θ and the internal geometry of the tract.

Acknowledgment

This work was generated with the assistance of LLMs.

References

[1] Aman Chawla. *On Axon-Axon Interaction via Currents and Fields*. PhD thesis, University of South Florida, 2017.

[2] Aman Chawla, Salvatore Morgera, and Arthur Snider. On axon interaction and its role in neurological networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2019.

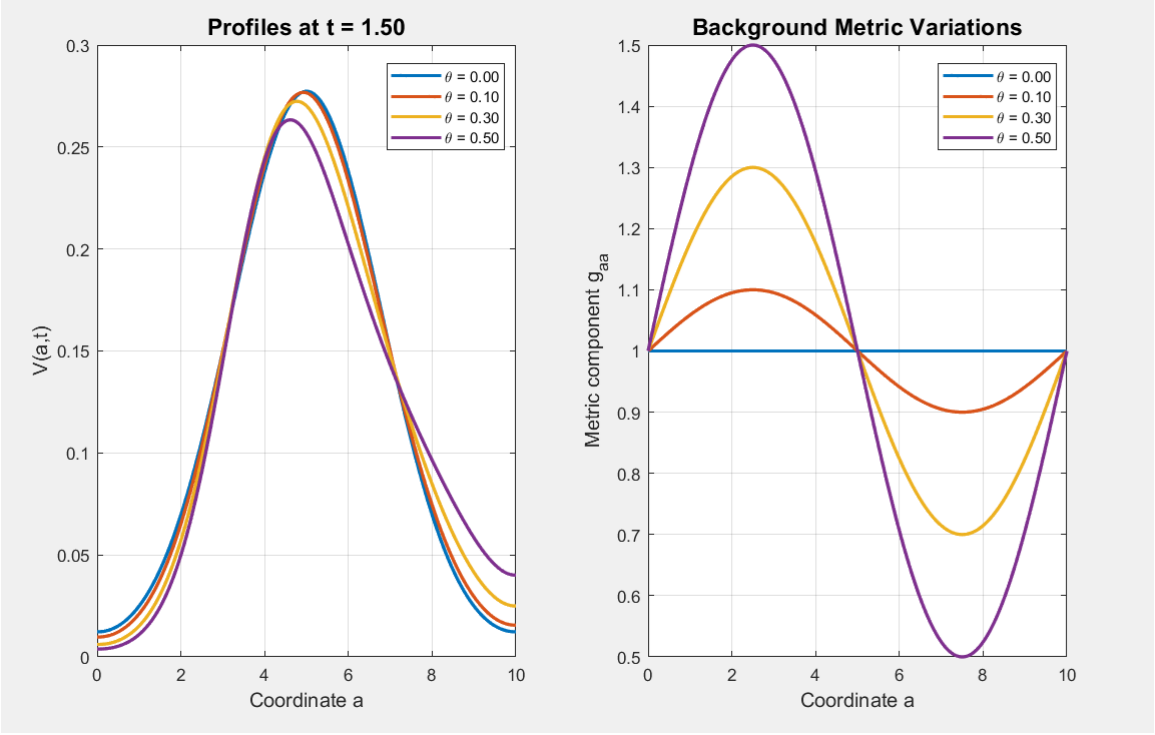

Figure 2: The code's output, part 2.

[3] Aman Chawla, Salvatore D. Morgera, and Arthur D. Snider. Fields, geometry, and their impact on axon interaction. *Journal of Applied Mathematics and Physics*, 9(4):751–778, 2021.

[4] Serge Lang. *Fundamentals of differential geometry*, volume 191. Springer Science & Business Media, 2012.