The spin-transport documentation

Rico A.R. Picone*

Feb 18, 2018

Abstract

The *spin-transport* software (GitHub) is for the dynamic simulation of bulk spin transport—diffusion and separation—in solid media. The project is open-source and still in development.

Contents

1 spin-transport: introduction

This repository contains the (developing) open-source code for simulating bulk spin transport—diffusion and separation—in solid media. Multi-spin-species and magnetic resonance simulations are in development.

This is a Python and FEniCS project. FEniCS is used to numerically solve the spin transport governing partial differential equations.

End users of this project write Python code to interface with FEniCS.

*Email: rpicone@stmartin.edu. Department of Mechanical Engineering, Saint Martin's University.

1.1 Installation

One must first have a working installation of FEniCS. This README assumes the use of Docker for installation, which is documented here.

Then clone this repository to the host machine.

1.2 Workflow

The FEniCS docs have a section on workflow. There are many ways to instantiate these good practices, but if you're using a nix system, the following may be the easiest.

With the cloned `spin-transport` repository as your working directory, create a link in your path to **spin-transport**'s `fenics` executable bash script.

```
ln fenics /usr/local/bin
```

Now a FEniCS Python script `foo.py` can be started with the command `fenics foo.py` **from the host** instead of manually starting it from a Docker container. This has several advantages, including that there is no need to move scripts into the container and that the complicated syntax need not be remembered.

1.3 Testing the installation

To verify that everything is installed correctly, run the Poisson equation demo `ft01_poisson.py` (source) in your container.

If you installed the `fenics` bash script per the instructions above, you can use the following command (working directory: `spin-transport`).

```
$ fenics ft01_poisson.py
```

If everything is working fine, the output should look something like the following.

```
$ fenics ft01_poisson.py
Calling DOLFIN just-in-time (JIT) compiler, this may take some time.
--- Instant: compiling ---
Calling FFC just-in-time (JIT) compiler, this may take some time.
Calling FFC just-in-time (JIT) compiler, this may take some time.
Calling FFC just-in-time (JIT) compiler, this may take some time.
Solving linear variational problem.
*** Warning: Degree of exact solution may be inadequate for accurate
        result in errornorm.
Calling FFC just-in-time (JIT) compiler, this may take some time.
Calling FFC just-in-time (JIT) compiler, this may take some time.
  Ignoring precision in integral metadata compiled using quadrature
  representation. Not implemented.
Calling FFC just-in-time (JIT) compiler, this may take some time.
Calling FFC just-in-time (JIT) compiler, this may take some time.
```

```
Calling FFC just-in-time (JIT) compiler, this may take some time.
error_L2  = 0.00823509807335
error_max = 1.33226762955e-15
```

The directory `spin-transport/poisson` should have been created and should contain two files: `solution.pvd` and `solution000000.vtu`. These files contain the solution data.

1.4 Acknowledgement

This work is supported by a grant from the Army Research Office, Materials Science Division under grant proposal **Nanoscale Spin Hyperpolarization and Imaging** with PI John Marohn, PhD.

1.5 Contributors

This project stems from a collaboration among three institutions:

- Cornell University,
- Saint Martin's University, and the
- University of Washington.

The lead contributor to this project is Rico Picone, PhD of Saint Martin's University, co-PI on the ARO grant. Other contributors include John Marohn, PhD (Cornell, PI), John A. Sidles, PhD (Washington), Joseph L. Garbini, PhD (Washington), and Corinne Isaac (Cornell).

2 Short theoretical introduction

The simulation is based on the two spin-species transport equations presented by Picone [4]. These equations present the transport—diffusion and separation—of three conserved quantities represented by the vector-valued function $\boldsymbol{\rho}$ of time and a single spatial dimension, taken to be the direction of a background magnetic field B we call *longitudinal*. The three conserved quantities are represented in the "polarization thermodynamic covector basis" $\mathfrak{e}$ by the components [4]

- ρ_1: a dimensionless dipole-energy density,
- ρ_2: longitudinal nuclear polarization, and
- ρ_3: longitudinal electron polarization.

Several parameters require definition and are summarized in the following list (note that a "~" over a function denotes that it is a function of the dimensionless spatial coordinate). See [4] for details.

- B_d: maximum dipole-dipole magnetic field.
- γ_2, γ_3: gyromagnetic ratio for nuclei and electrons, respectively.
- $\overline{\gamma}$: dimensionless ratio γ_3/γ_2.
- Γ_2, Γ_3: transport coefficients for nuclear and electron polarization, respectively.
- $\overline{\Gamma}$: dimensionless ratio Γ_3/Γ_2.
- Δ_2, Δ_3: constant spin densities in solid medium for nuclei and electrons, respectively.
- $\overline{\Delta}$: dimensionless ratio Δ_3/Δ_2.
- $\overline{r}$: dimensionless spatial coordinate $(\partial_r B(r)|_{r=0}/B_d)r$.
- $\overline{t}$: dimensionless time $\Gamma_2(\partial_r B(r)|_{r=0}/B_d)^2 t$.
- $\overline{B}$: ratio of magnetic fields $\partial_{\overline{r}}\tilde{B}/B_d$.
- $\overline{c}$: ratio of ratios $\overline{B}(1+\overline{\Delta})/(1+\overline{\gamma}\overline{\Delta})$.

Two-species Bloch-transport equations. The preferred form of the continuity equation is in the polarization basis with the dimensionless spatial coordinate and time variable, which we call the *two-species magnetization transport equations*:

$$\begin{aligned}
\partial_{\overline{t}}\rho_1 &= -\frac{\overline{c}^2}{1+\overline{\Delta}}\left(\left(1-\rho_2^2\right)+\overline{\Gamma\Delta}\overline{\gamma}^2\left(1-\rho_3^2\right)\right)\text{arctanh }\rho_1 + \\
&\quad -\frac{\overline{c}}{1+\overline{\Delta}}\left(\partial_{\overline{r}}\rho_2-\overline{\Gamma\Delta}\overline{\gamma}\partial_{\overline{r}}\rho_3\right)+\left(1+\overline{\Gamma}\right)\partial_{\overline{r}}^2\rho_1 \qquad (1)\\
\partial_{\overline{t}}\rho_2 &= \partial_{\overline{r}}\left(\overline{c}\left(1-\rho_2^2\right)\text{arctanh }\rho_1\right)+\partial_{\overline{r}}^2\rho_2 \qquad (2)\\
\partial_{\overline{t}}\rho_3 &= -\overline{\Gamma}\overline{\gamma}\partial_{\overline{r}}\left(\overline{c}\left(1-\rho_3^2\right)\text{arctanh }\rho_1\right)+\overline{\Gamma}\partial_{\overline{r}}^2\rho_3. \qquad (3)
\end{aligned}$$

Bloch equation dynamics for a each species and $T_1 \gg T_2$ can be approximated without the transverse dynamics as [3, 1]

$$\begin{aligned}
\partial_{\overline{t}}\rho_2 &= \frac{\rho_{20}}{\overline{T}_{1p}}-\frac{\rho_2}{\overline{\tau}_p} \qquad (4)\\
\partial_{\overline{t}}\rho_3 &= \frac{\rho_{30}}{\overline{T}_{1e}}-\frac{\rho_3}{\overline{\tau}_e}. \qquad (5)
\end{aligned}$$

where, for each species,

$$\overline{\tau} = \frac{1}{\frac{1}{\overline{T}_1} + \frac{\overline{T}_2 \omega_1^2}{1+\overline{T}_2^2 \tilde{\delta}^2}} \tag{6}$$

where it is important to use dimensionless time constants and where we define (again, for each species)

$$\omega_1 = -\gamma B_1 \quad \text{and} \quad \tilde{\delta} = \gamma B_d \overline{r}. \tag{7}$$

The Bloch equation dynamics can be combined with the transport dynamics of Equations 1 – 3 such that the *two-species Bloch-transport* equations can be written

$$\begin{aligned} \partial_{\overline{t}} \rho_1 = &-\frac{\overline{c}^2}{1+\overline{\Delta}} \left(\left(1 - \rho_2^2 \right) + \overline{\Gamma \Delta} \overline{\gamma}^2 \left(1 - \rho_3^2 \right) \right) \text{arctanh } \rho_1 + \\ &- \frac{\overline{c}}{1+\overline{\Delta}} \left(\partial_{\overline{r}} \rho_2 - \overline{\Gamma \Delta} \overline{\gamma} \partial_{\overline{r}} \rho_3 \right) + \left(1 + \overline{\Gamma} \right) \partial_{\overline{r}}^2 \rho_1 \end{aligned} \tag{8}$$

$$\partial_{\overline{t}} \rho_2 = \partial_{\overline{r}} \left(\overline{c} \left(1 - \rho_2^2 \right) \text{arctanh } \rho_1 \right) + \partial_{\overline{r}}^2 \rho_2 + \frac{\rho_{20}}{\overline{T}_{1p}} - \frac{\rho_2}{\overline{\tau}_p} \tag{9}$$

$$\partial_{\overline{t}} \rho_3 = -\overline{\Gamma} \overline{\gamma} \partial_{\overline{r}} \left(\overline{c} \left(1 - \rho_3^2 \right) \text{arctanh } \rho_1 \right) + \overline{\Gamma} \partial_{\overline{r}}^2 \rho_3 + \frac{\rho_{30}}{\overline{T}_{1e}} - \frac{\rho_3}{\overline{\tau}_e}. \tag{10}$$

This system of equations is that which is explored, numerically, by the *spin-transport* software.

3 Variational formulation for mixed boundary conditions

The FEniCS solver requires the system of equations be expressed in *variational form*, also called *weak form*.

Furthermore, the system must be discretized in time. The backward Euler method yields time derivative approximations, for a time-index n and time-step T,

$$\partial_{\overline{t}} \rho_i = \frac{\rho_i^{n+1} - \rho_i^n}{T}. \tag{11}$$

For variational form, each $\partial_{\overline{r}}^2 \rho_i$ term is integrated by parts—with test functions $v_i(\overline{r})$ over the spatial domain Ω with boundary $\partial\Omega$, subsets of which, Γ_D

and Γ_N, satisfy *Dirichlet* and *Neumann* boundary conditions $\rho_i = \rho_{i0}$ on Γ_D and $-\partial_{\bar{r}}\rho_i = G_i$ (for no-flow $G_i = 0$) on Γ_N—as follows [2, § 4.1.1–4.1.2]:

$$\int_\Omega \partial_{\bar{r}}^2 \rho_i = -\int_\Omega \partial_{\bar{r}}\rho_i \cdot \partial_{\bar{r}} \nu_i \, d\bar{r} + \int_{\partial\Omega} \nu_i \partial_{\bar{r}} \rho_i \, ds \tag{12}$$

$$= -\int_\Omega \partial_{\bar{r}}\rho_i \cdot \partial_{\bar{r}} \nu_i \, d\bar{r} - \int_{\Gamma_N} \nu_i G_i \, ds. \tag{13}$$

This yields the weak formulation

$$\begin{aligned}
&\int_\Omega \Bigg(T^{-1}(\rho_1^{n+1} - \rho_1^n)\nu_1 + (1+\bar{\Gamma})\partial_{\bar{r}}\rho_1^{n+1} \cdot \partial_{\bar{r}}\nu_1 \\
&\qquad + \frac{\bar{c}^2}{1+\bar{\Delta}} \left(\left(1 - (\rho_2^{n+1})^2\right) + \overline{\Gamma\Delta}\bar{\gamma}^2 \left(1 - (\rho_3^{n+1})^2\right) \right) \nu_1 \operatorname{arctanh} \rho_1^{n+1} + \\
&\qquad - \frac{\bar{c}}{1+\bar{\Delta}} \left(\partial_{\bar{r}}\rho_2^{n+1} - \overline{\Gamma\Delta}\bar{\gamma}\partial_{\bar{r}}\rho_3^{n+1} \right) \nu_1 \Bigg) d\bar{r} + \\
&\int_\Omega \Bigg(T^{-1}(\rho_2^{n+1} - \rho_2^n)\nu_2 + \partial_{\bar{r}}\rho_2^{n+1} \cdot \partial_{\bar{r}}\nu_2 \\
&\qquad \left(-\bar{c}\frac{1-(\rho_2^{n+1})^2}{1-(\rho_1^{n+1})^2}\partial_{\bar{r}}\rho_1^{n+1} + 2\bar{c}\rho_2^{n+1} \operatorname{arctanh} \rho_1^{n+1} \partial_{\bar{r}}\rho_2^{n+1} + \frac{\rho_2^{n+1}}{\bar{\tau}_p} \right) \nu_2 \Bigg) d\bar{r} + \\
&\int_\Omega \Bigg(T^{-1}(\rho_3^{n+1} - \rho_3^n)\nu_3 + \bar{\Gamma}\partial_{\bar{r}}\rho_3^{n+1} \cdot \partial_{\bar{r}}\nu_3 \\
&\qquad \left(-\bar{c}\frac{1-(\rho_3^{n+1})^2}{1-(\rho_1^{n+1})^2}\partial_{\bar{r}}\rho_1^{n+1} + 2\bar{c}\rho_3^{n+1} \operatorname{arctanh} \rho_1^{n+1} \partial_{\bar{r}}\rho_3^{n+1} + \frac{\rho_3^{n+1}}{\bar{\tau}_e} \right) \nu_3 \Bigg) d\bar{r} + \\
&-\int_\Omega \left(\frac{\rho_{20}}{\bar{T}_{1p}}\nu_2 + \frac{\rho_{30}}{\bar{T}_{1e}}\nu_3 \right) d\bar{r} + \int_{\Gamma_N} \left(\nu_1 G_1 + \nu_2 G_2 + \nu_3 G_3 \right) ds = 0.
\end{aligned} \tag{14}$$

References

[1] Jean-Philippe Grivet. Simulation of Magnetic Resonance Experiments. *American Journal of Physics*, 61(12):1133–1139, 1993.

[2] H.P. Langtangen and A. Logg. *Solving PDEs in Python: the FEniCS Tutorial I*. Simula SpringerBriefs on Computing. Springer, 2017.

[3] Rico A. R. Picone. Investigating the Effects of Polarizing Diffusion in Magnetic Resonance Force Microscopy, 2010.

[4] Rico A. R. Picone. Separative Magnetization Transport: Theory, Model, and Experiment, 2014.