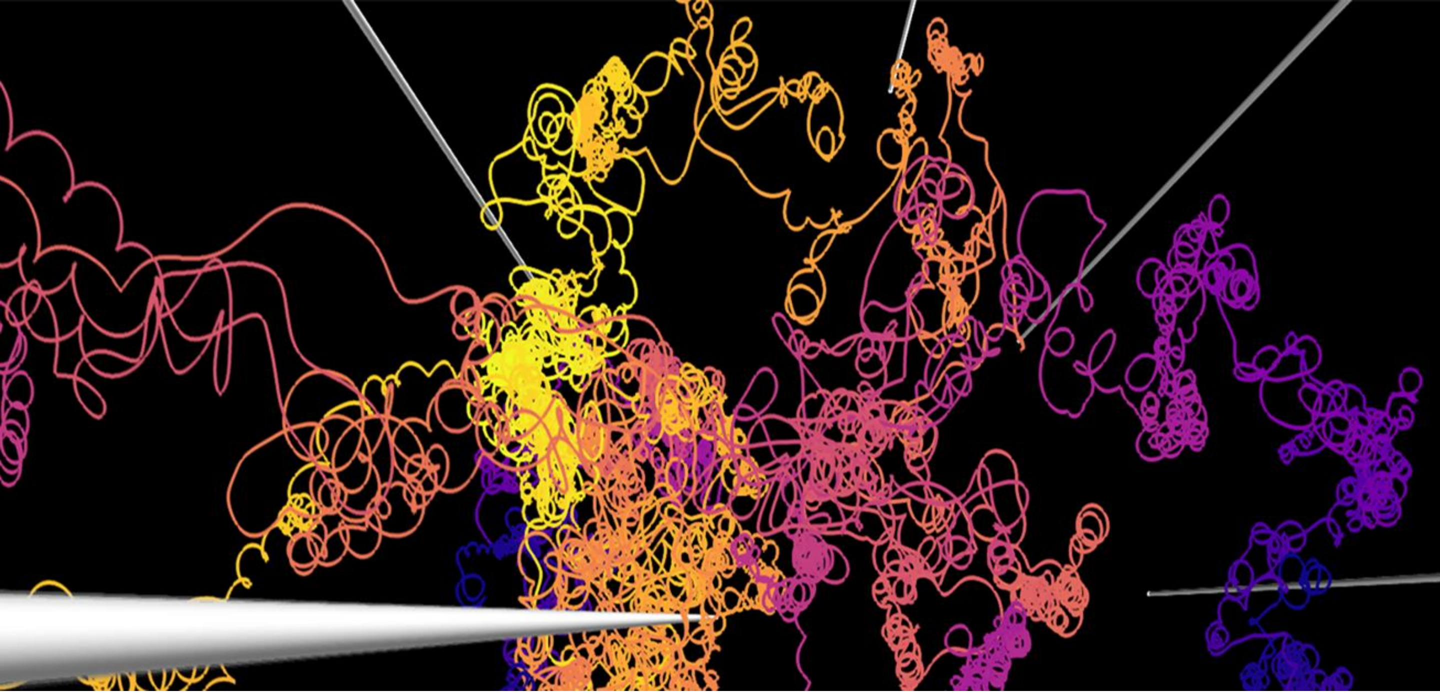


RAPP Center and Leopold Franzens Universität Innsbruck

Progress of the momentum diffusion module

Lukas Merten

CRPropa Workshop 2019 – DESY-Zeuthen

Contents

TRANSPORTEQUATION

Transport Equation

Spatial Diffusion — Sources

$$\frac{\partial n(\vec{r},p,t)}{\partial t} + \vec{u}\cdot\nabla n = \nabla\cdot(\hat{\kappa}\nabla n) + \frac{p}{3}(\nabla\cdot\vec{u})\frac{\partial n}{\partial p} + S$$

Advection — Adiabatic effects

Partial (Fokker-Planck) Differential Equation (PDE) for particle density n

Equivalence

Stochastic Differential Equation(SDGs)

$$\mathrm{d}\vec{x} = \underline{\vec{u}dt} + \underline{\widehat{D}\mathrm{d}\vec{w}}$$

$$\mathrm{d}p = \underline{-{}^{p}/_{3}\,(\nabla\cdot\vec{u})\mathrm{dt}}$$

Assumption:

- Isotropic CR density ($n \neq n(\vec{p})$)
- No CR feedback on magnetic fields

From here on:

- Solving the SDE in the local magnetic field line frame
- Neglect momentum diffusion

Comparison grid vs. SDE

Grid-based:
GalProp, DRAGON, PICARD, …

Advantages
- ✓ Implement collective behavior
- ✓ Tested and well understood
- ✓ (nearly) complete

- ✓ PICARD: Explicit stationary solver

Disadvantages
- Huge RAM
- Not possible to reweight
- No information on single particles

SDE-based:
CRPropa, Kopp+ (´12), Miyake+ (´14), …

Advantages
- ✓ Scales linearly with number of processors
- ✓ Reweighting is possible
- ✓ Not restricted to grid
- ✓ Backtracking possible
- ✓ Discontinuities can be handled

Disadvantages
- Averaging of results necessary
  → Many pseudo particles
- Not all interactions implemented yet
- Conceptually more complicated

Numerical integration scheme

New modules

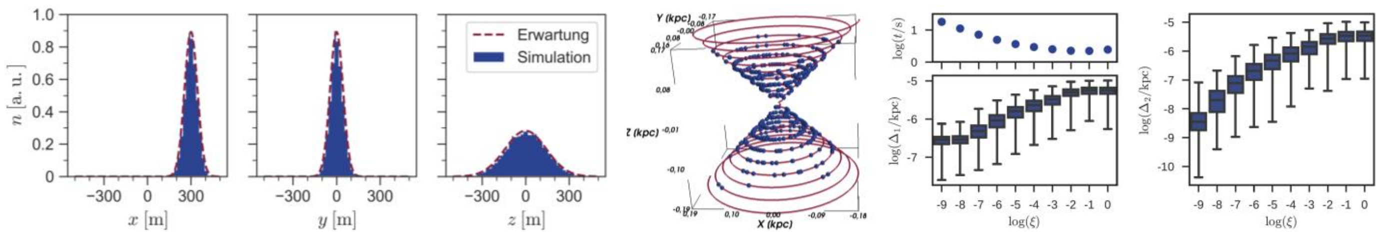
- DiffusionSDE
- AdiabaticCooling
- AdvectionField
- Homogeneous, Spherical Shock, etc.

Numerical solver: Euler-Maruyama Integration

$$x_{n+1} - x_n = (u_x\vec{e}_x + u_y\vec{e}_y + u_z\vec{e}_z)\cdot h$$
$$+\left(\sqrt{2\kappa_\parallel}\eta_\parallel\vec{e}_\mathrm{t} + \sqrt{2\kappa_\perp}\eta_{\perp,1}\vec{e}_\mathrm{n} + \sqrt{2\kappa_\perp}\eta_{\perp,2}\vec{e}_\mathrm{b}\right)\cdot\sqrt{h}$$
$$p_{n+1} - p_n = -{}^{p_n}/_{3}\,(\nabla\cdot\vec{u})\cdot h$$

Validation I: Homg. magn. field $\vec{B} = B_0\vec{e}_z$, wind $\vec{u} = u_0\vec{e}_x$ and aniso. Diffusion $\epsilon = 0{,}1$

Validation II: Spiral magn. Field, no wind and parallel Diffusion $\epsilon = 0$ only



Stat. Solution / Finite Source duration

For the Galactic CR distribution often a stationary solution ($^{\partial n}/_{\partial t} = 0$) or finite source duration, e.g., ($S = S_0 \cdot \Theta(t - t_0)\Theta(\mathrm{t}_1 - \mathrm{t}), \mathrm{t}_1 > \mathrm{t}_0$) is of interest

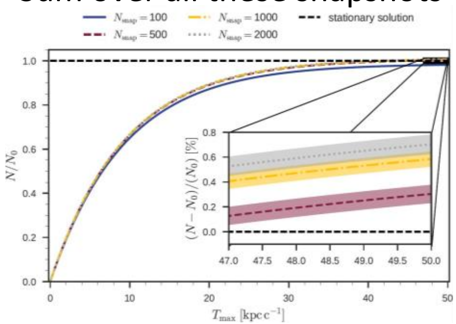
Problem

CRPropa does only simulate a bursting source $S \propto \delta(t - t_0)$

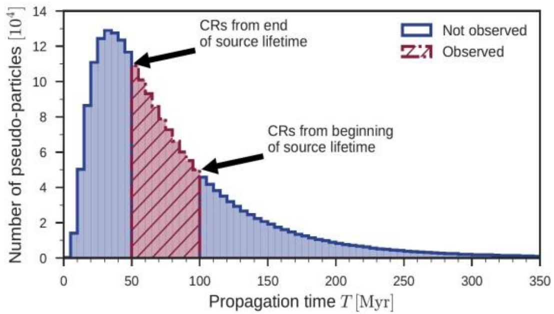
Solution

Clever data taking and reweighting of simulation results

- Register all CRs in the simulation volume several times
- Sum over all these snapshots



- Continuously register all CR at a given observer
- Sum over time ordered subsets



MOMENTUM DIFFUSION

Transport Equation

$$\frac{\partial n(\vec{r},p,t)}{\partial t} + \vec{u}\cdot\nabla n = \nabla\cdot(\hat{\kappa}\nabla n) + \boxed{\frac{1}{p^2}\frac{\partial}{\partial p}\left(p^2\kappa_{pp}\frac{\partial n}{\partial p}\right)} + \frac{p}{3}(\nabla\cdot\vec{u})\frac{\partial n}{\partial p} + S$$

Momentum Diffusion

$$\mathrm{d}\vec{x} = \vec{u}dt + \widehat{D}\mathrm{d}\vec{w}$$

$$\mathrm{d}p = -{}^{p}/_{3}\,(\nabla\cdot\vec{u})\mathrm{dt} + \underline{\left({}^{\partial\kappa_{pp}}/_{\partial p} - {}^{2}/_{p}\,\kappa_{pp}\right)\mathrm{dt} + D_{\mathrm{pp}}\mathrm{d}w_{\mathrm{p}}}$$

$$x_{n+1} - x_n = (u_x\vec{e}_x + u_y\vec{e}_y + u_z\vec{e}_z)\cdot h$$
$$+\left(\sqrt{2\kappa_\parallel}\eta_\parallel\vec{e}_{\mathrm{t}} + \sqrt{2\kappa_\perp}\eta_{\perp,1}\vec{e}_{\mathrm{n}} + \sqrt{2\kappa_\perp}\eta_{\perp,2}\vec{e}_{\mathrm{b}}\right)\cdot\sqrt{h}$$
$$p_{n+1} - p_n = \left(-{}^{p_n}/_{3}\,(\nabla\cdot\vec{u}) + {}^{\partial\kappa_{pp}}/_{\partial p} - {}^{2\cdot\kappa_{pp}}/_{p}\right)\cdot h$$
$$+\left(\sqrt{2\kappa_{pp}}\eta_p\right)\cdot\sqrt{h}$$

Relation of spatial and momentum diffusion

1. The momentum diffusion is generally defined as: $\kappa_{pp} = \frac{\langle \Delta p^2 \rangle}{\tau}$
   (see, e.g., TGK formulation [Taylor 1922, Green 1951, Kubo 1958])
2. The mean momentum transfer for diffusion in Alfvén waves per scattering process is: $\Delta p \propto v_A p$
3. The scattering time is connected to the spatial diffusion $\tau \propto \hat{\kappa}(p)$

$$\kappa_{pp} \propto \frac{v_A^2 p^2}{\kappa(p)}$$

This allows to easily implement a model for momentum diffusion, when the spatial diffusion is established.

STATUS

Work in progress…

- First implementation of momentum diffusion is done
- Up to now it is planned to be part of the DiffusionSDE-module
  - Make use of the existing structure
  - Avoid code duplication
  - Make use of calculation that have to be done anyway

  - Makes the module even longer
  - Maybe hard to debug and maintain

- There is version on my GitHub-page
  - WARNING: Not tested extensively, yet.

The process routine

```
void MomentumDiffusion::process(Candidate *c) const {

  double p = c->current.getEnergy()/c_light; // Note we use E=p/c (relativistic limit)
  double rig = p*c_light / c->current.getCharge();
  double dt = c->getCurrentStep() / c_light;

  std::cout <<dt<<"\n";
  double eta =  Random::instance().randNorm();
  double domega = eta * sqrt(dt);


  double AScal = calculateAScalar(rig, p);
  double BScal = calculateBScalar(rig, p);

  double dp = AScal * dt + BScal * domega;
  std::cout <<dp<<"\n";
  c->current.setEnergy((p + dp)*c_light);

  //c->limitNextStep(limit * p / ((AScal + BScal/sqrt(dt)) * c_light));
  c->limitNextStep(limit * p / ((AScal + BScal/sqrt(dt)) * c_light)); //Check for the factor c_light

}
```

The physics behind

```
double MomentumDiffusion::calculateBScalar(double rig, double p) const{

        double Dxx = scale * 6.1e24 * pow((std::abs(rig) / 4.0e9), alpha);
   double Dpp = ( 4*vA*vA*p*p ) / ( 3*alpha*(4-alpha*alpha)*(4-alpha) ) / Dxx; // Astroparticle Physics: Theory and Phenomenology, G. Sigl,
        double BScal = sqrt( 2  * Dpp);
        return BScal;

}

// What is the physical interpretetation of this term? 7/27/19 LM
double MomentumDiffusion::calculateAScalar(double rig, double p) const {

        double Dxx = scale * 6.1e24 * pow((std::abs(rig) / 4.0e9), alpha);
   double Dpp = ( 4*vA*vA*p*p ) / ( 3*alpha*(4-alpha*alpha)*(4-alpha) ) / Dxx;
        double partialDpp = (2 - alpha) / p * Dpp; //check the sign: Should be correct 7/27/19 LM

        double AScal = partialDpp -2. / p * Dpp; //=-alpha / p * Dpp
      return AScal;

}
```

Lukas.Merten@rub.de

SUMMARY / OUTLOOK

Open problems / Solutions

Implement momentum diffusion as part of Diffusion SDE

Make it flexible

- Connected to implemention of $\left(\frac{\delta b}{B}\right)$-dependence of the spatial diffusion tensor

Validation and testing

- Compare to earlier results, e.g., by Tobias (arXiv:1612:03675v2)