

Chemically reactive flows in SPH: a basic overview

Paul Clark

Zentrum für Astronomie (ZAH) der Universität Heidelberg
Institut für Theoretische Astrophysik

Summary of the week

Today

Introduction to Seren (the SPH for this week) and basic ideas behind doing chemistry in fluid codes.

Tuesday

How to create a simple model of the ISM with H_2/H^+ chemistry and basic ISM heating and cooling

Wednesday

Ways to improve on the simple model: CO, better treatment of shielding, radiative transfer, treating optically-thick line-cooling.

Thursday

Overview of Pop III : overview of the main chemical processes, how it can go wrong(!), improving optically-thick line-cooling

Friday

SPH add-ons and improvements: sink particles, diffusion, new fixes.

Seren: an SPH code

- Written by David Hubber, and SPH expert (means “star” in Welsh!)
- It’s a new code (unlike the organic growths that most groups use), written in clear F90
- Has a flexible SPH implementation under the hood
- It is maintained, and can be accessed via GIT
- Has support for MPI/OpenMP
- It has several excellent code papers attached to it:

Hubber et al. 2011, A&A, 529, 27

Hubber et al. 2013, MNRAS, 430, 1599

Hubber et al. 2013, MNRAS, 430, 3261

Hubber et al. 2013, MNRAS, 432, 711

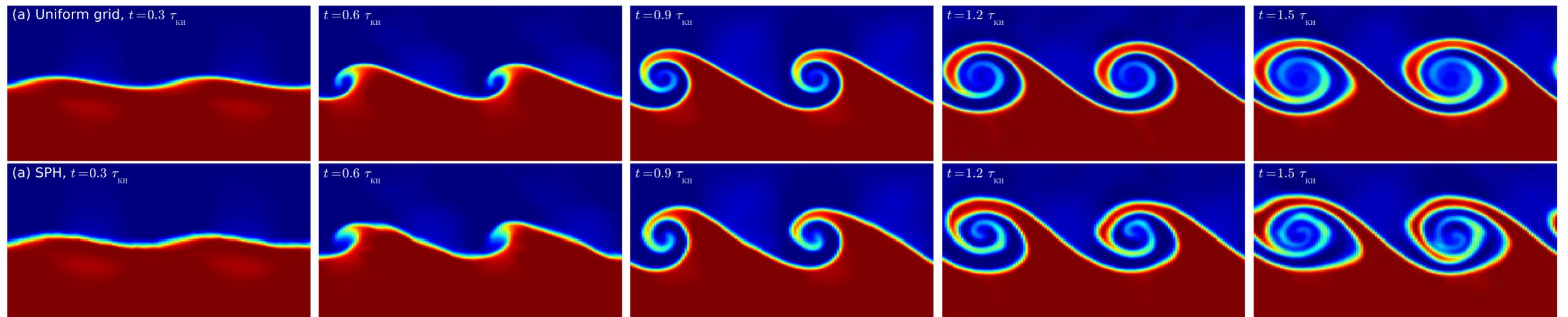
Inside Seren

- Several time-integration methods
- Individual particle timesteps (+ limiters)
- Different flavours of SPH:
 - different implementations (original/'grad-H'/entropy formalism)
 - several different smoothing kernels
 - various forms of viscosity
- Hybrid N-body + gas (4th order Hermite for N-body)
- Improved sink particle implementation
- Several modifications to SPH
- Approximations for radiative transfer:
 - Dust continuum RT approx (Stamatellos/Forgan)
 - Photoionisation (Bisbas)

Code comparison paper

- Compares SPH to AMR for a series of standard test-problems

MG (Van Loo et al. 2006)

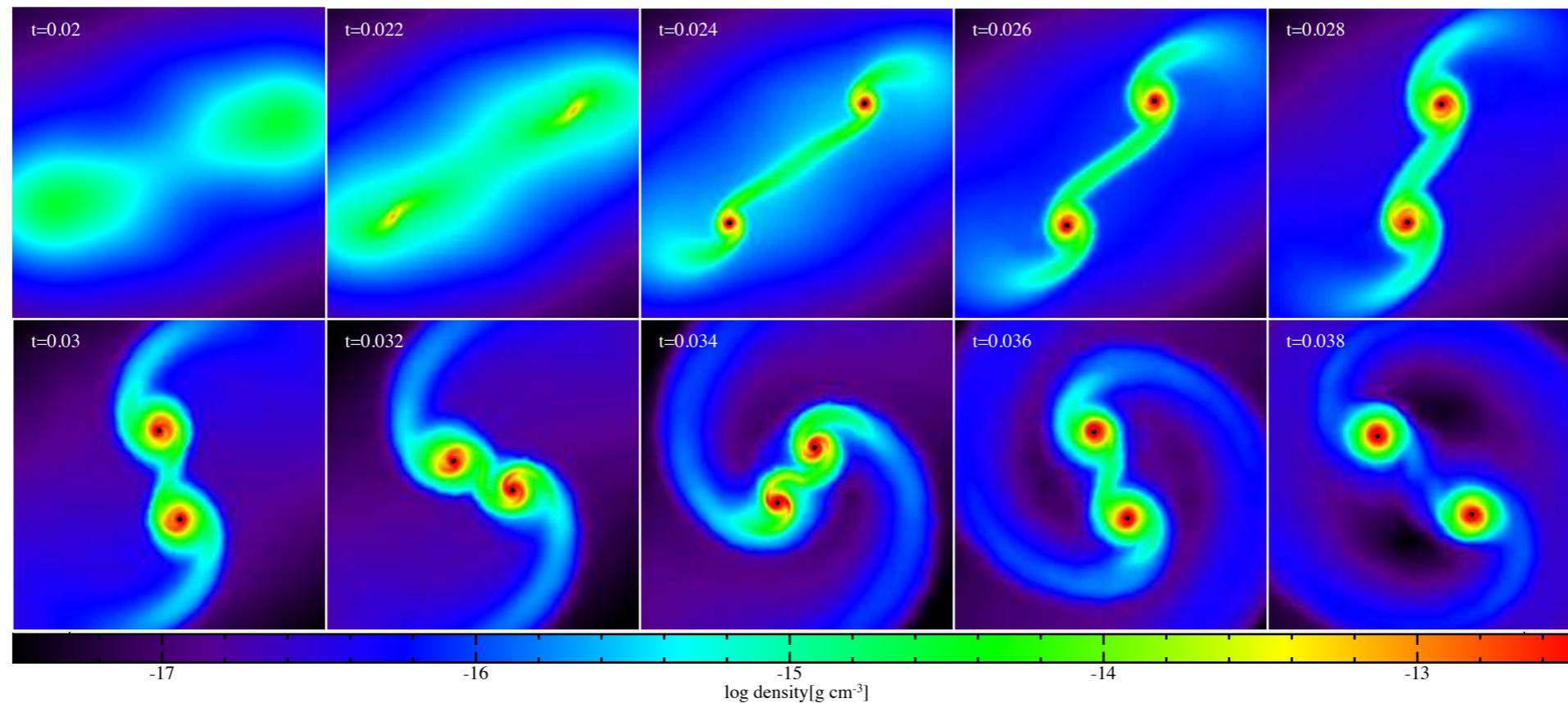


Seren

Hubber et al. (2013)

Improved sink particle algorithm

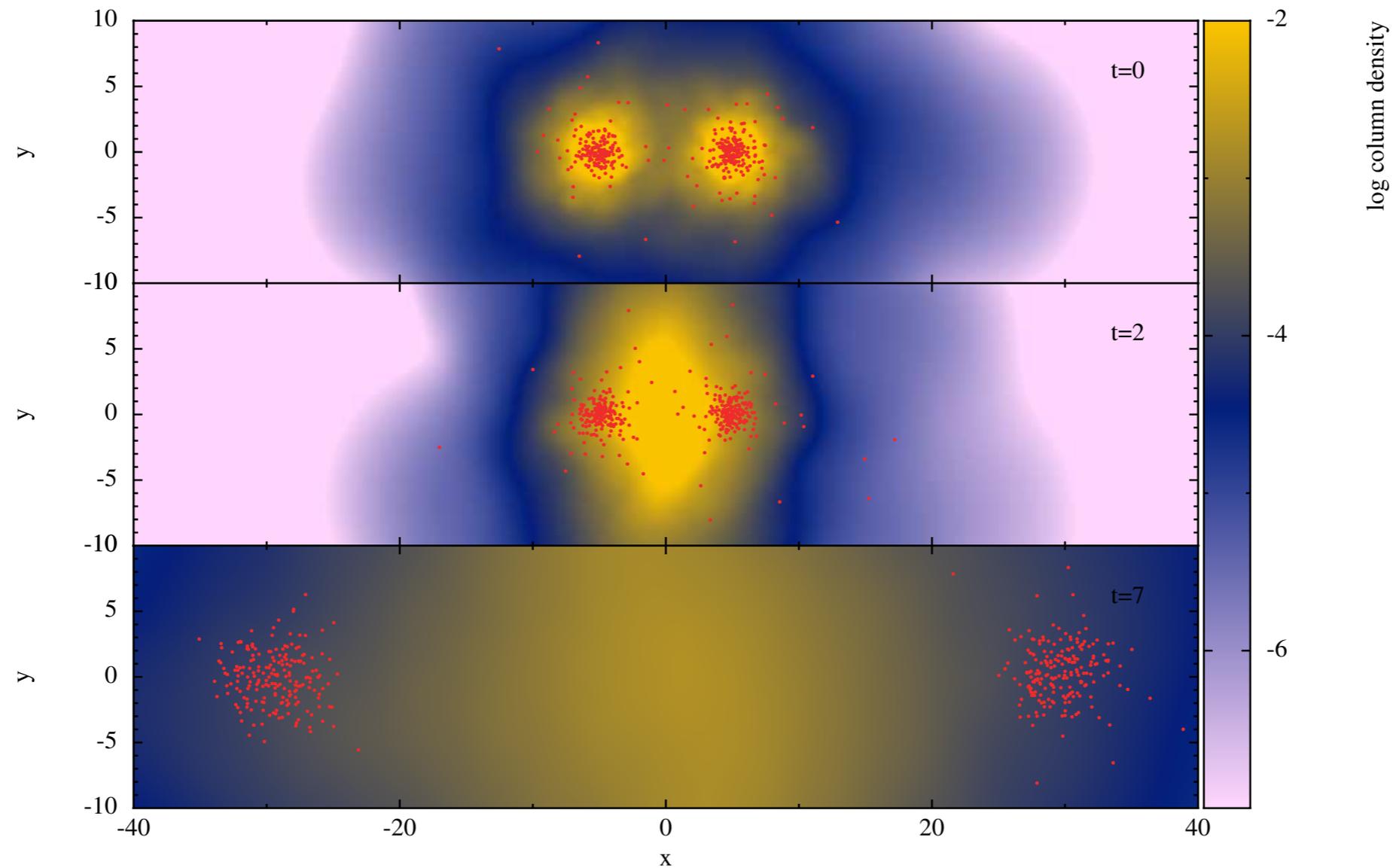
- “Boss-Bodenheimer test” (1979):



Hubber, Walch & Whitworth (2013)

Nbody-paper

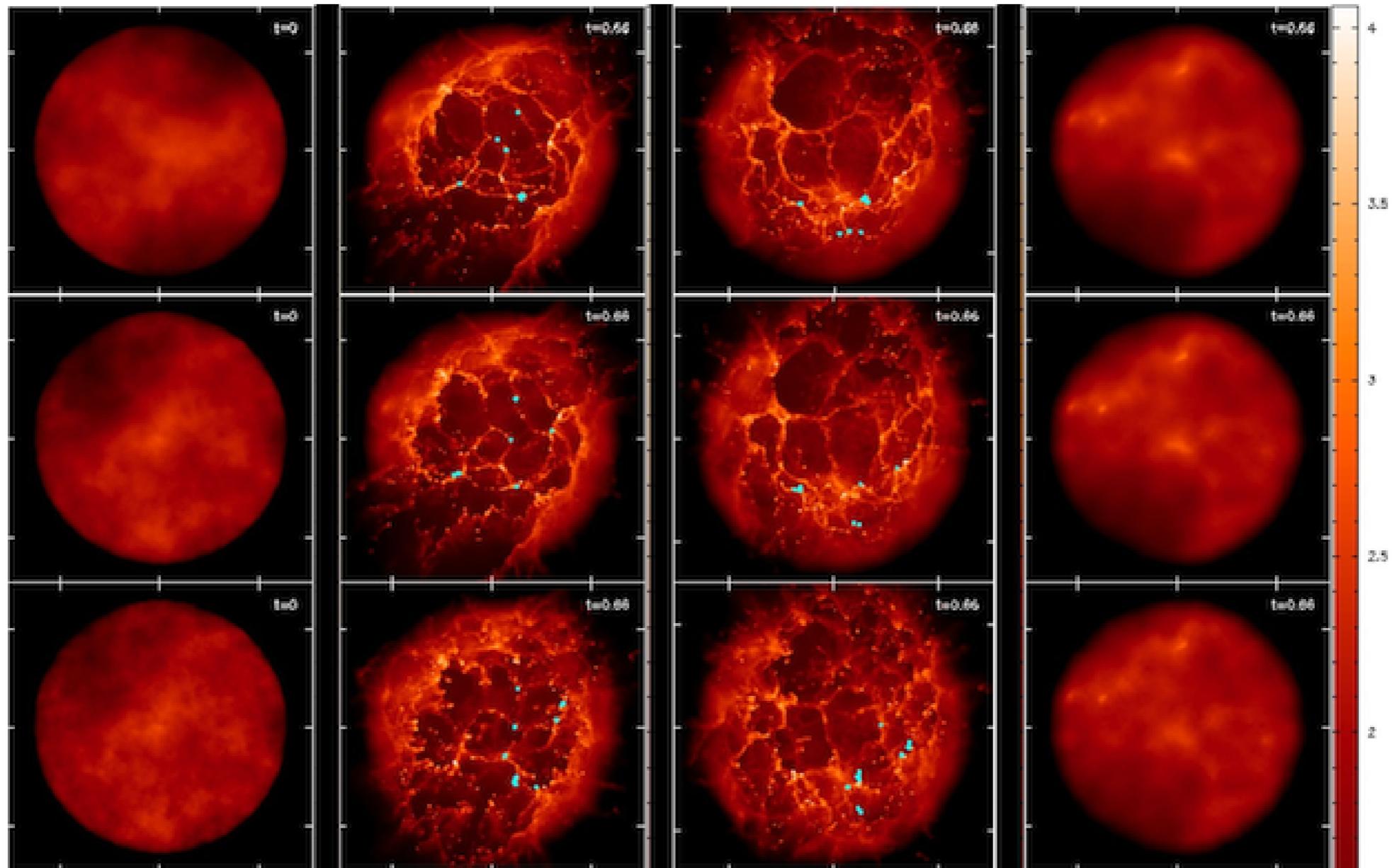
- Colliding plummer spheres with gas



Hubber et al. (2013)

Seren Science...

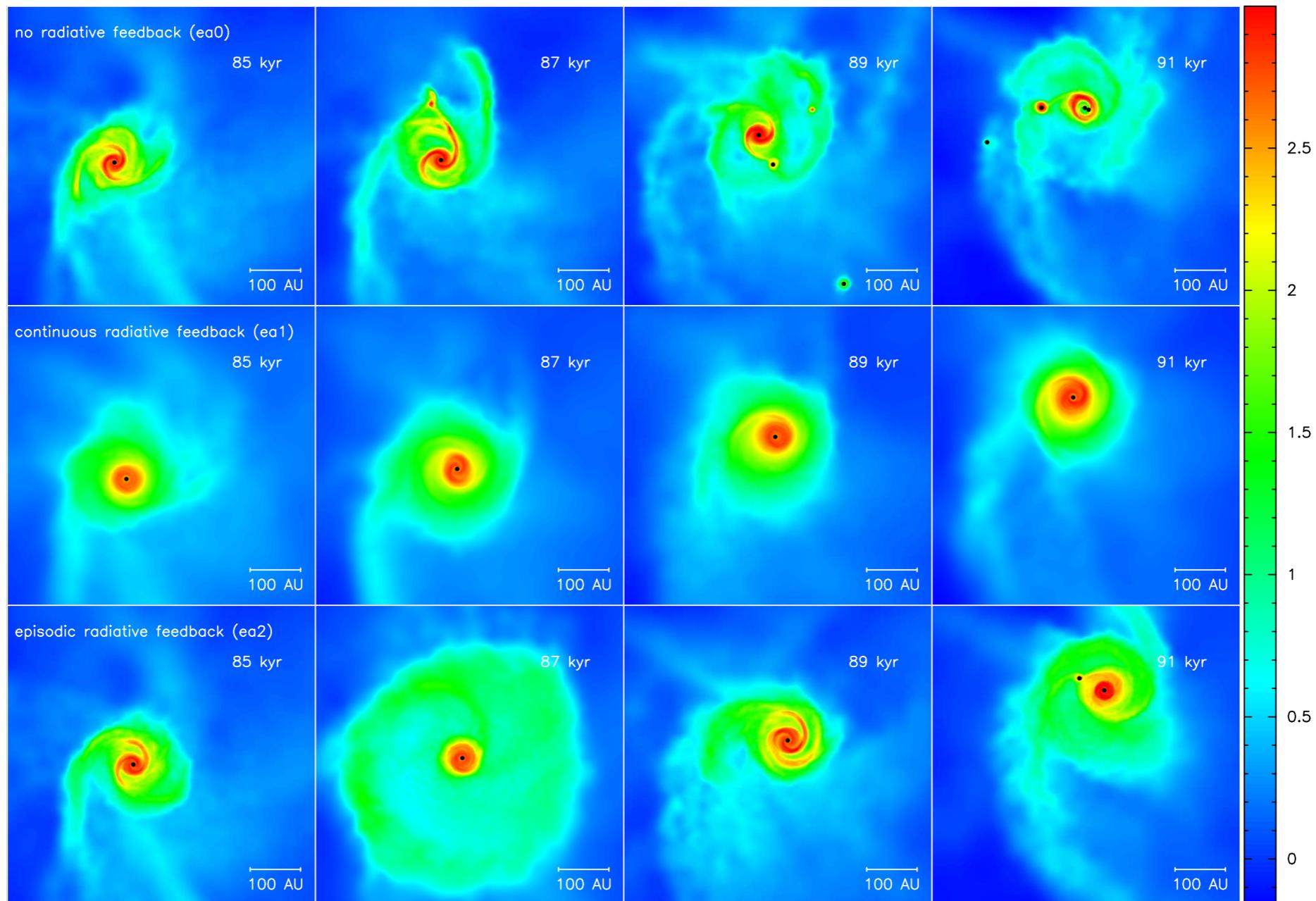
- Dispersal of molecular clouds via photoionisation



Walch et al. (2012)

Seren Science...

- Fragmentation of protostellar discs

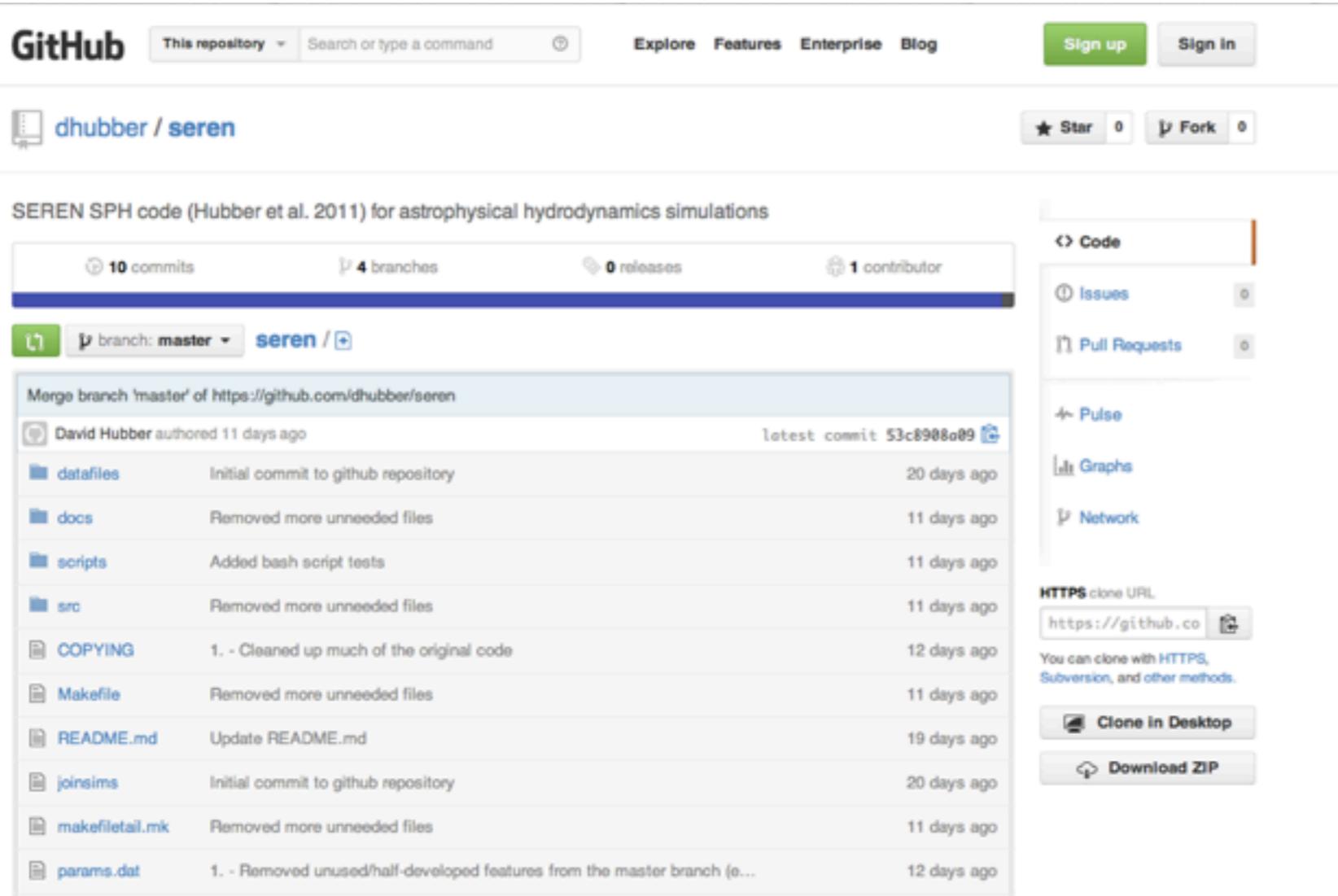


Stamatellos et al. (2012)

Obtaining the code

- Can be obtained via “git” at:

<https://github.com/dhubber/seren>



The screenshot shows the GitHub interface for the repository 'dhubber / seren'. At the top, there's the GitHub logo, a search bar, and navigation links like 'Explore', 'Features', 'Enterprise', and 'Blog'. There are 'Sign up' and 'Sign in' buttons. Below the repository name, it shows 'Star 0' and 'Fork 0'. The main content area displays the repository description: 'SEREN SPH code (Hubber et al. 2011) for astrophysical hydrodynamics simulations'. It also shows statistics: '10 commits', '4 branches', '0 releases', and '1 contributor'. A 'branch: master' dropdown is visible. The commit history is shown as a table with columns for file names, commit messages, and dates. The right sidebar contains navigation links for 'Code', 'Issues', 'Pull Requests', 'Pulse', 'Graphs', and 'Network'. At the bottom of the sidebar, there are options to 'Clone in Desktop' and 'Download ZIP'.

File	Commit Message	Time Ago
datafiles	Initial commit to github repository	20 days ago
docs	Removed more unneeded files	11 days ago
scripts	Added bash script tests	11 days ago
src	Removed more unneeded files	11 days ago
COPYING	1. - Cleaned up much of the original code	12 days ago
Makefile	Removed more unneeded files	11 days ago
README.md	Update README.md	19 days ago
joinsims	Initial commit to github repository	20 days ago
makefiletail.mk	Removed more unneeded files	11 days ago
params.dat	1. - Removed unused/half-developed features from the master branch (e...	12 days ago

- This talk will be posted online (pdf)

<http://www.ita.uni-heidelberg.de/~pcc/hipacc>

IO & ICs

- Seren has several formats for the IC files and snapshots.
- Easiest is simply to use the formatted ascii column format (called 'ascii' in the code).
- Can be read in simply with gnuplot / IDL etc.
- I've also uploaded an IC generator to hyades

`/home/pcc/setpartseren`

- A turbulent velocity cubes (128^3) can be found in:

`/home/pcc/turb_grid_128_cube_nat`

Goal of the week

- To include simple CO chemistry in Seren
- Try a few different simulations (turbulent cloud, colliding flows)
- Improve on the code!
 - Better treatment of shielding (Gnedin/Clark/other...)
 - Optimise the chemistry (much of it could be tabulated)
 - Live dust temperatures?

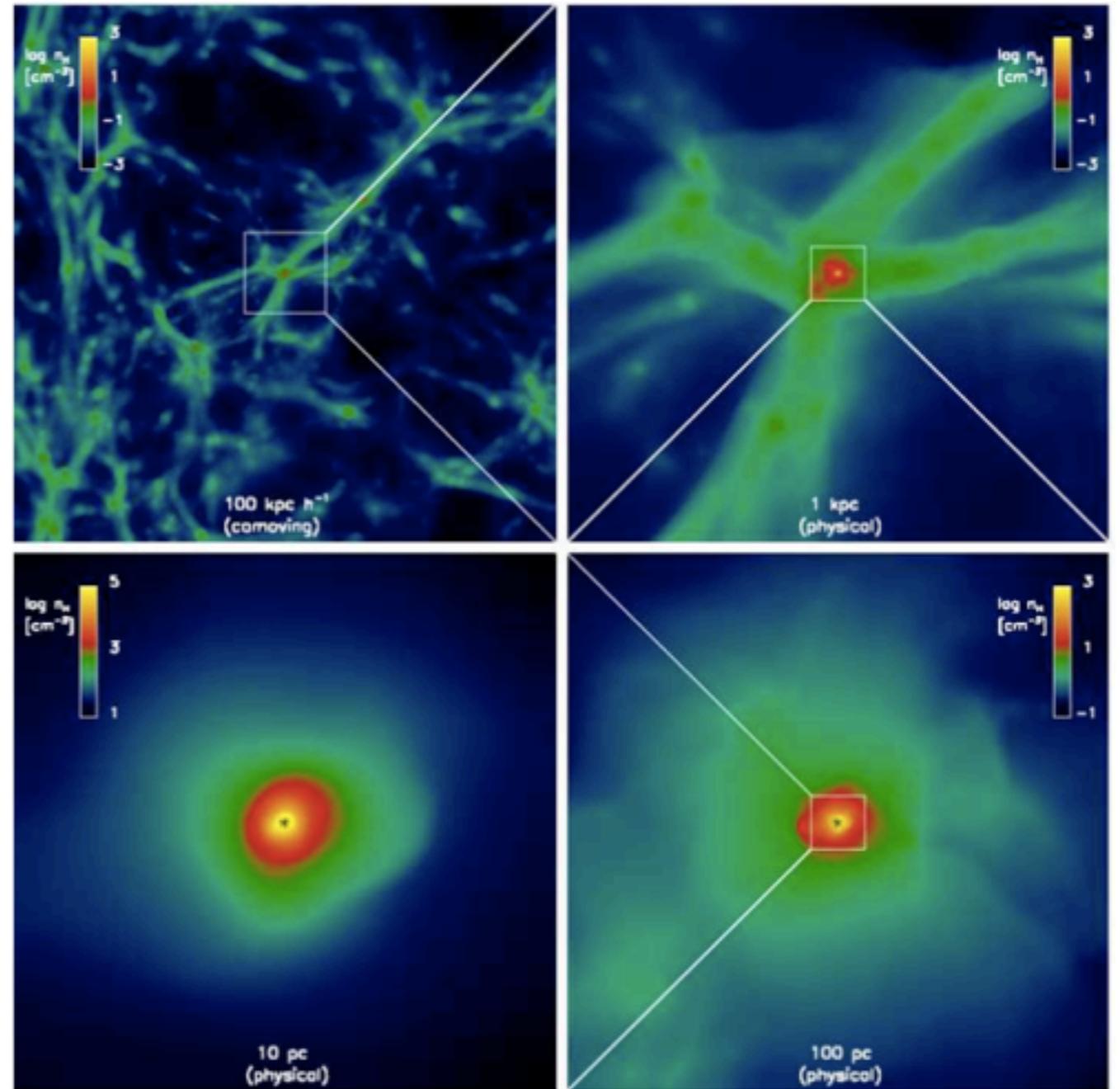
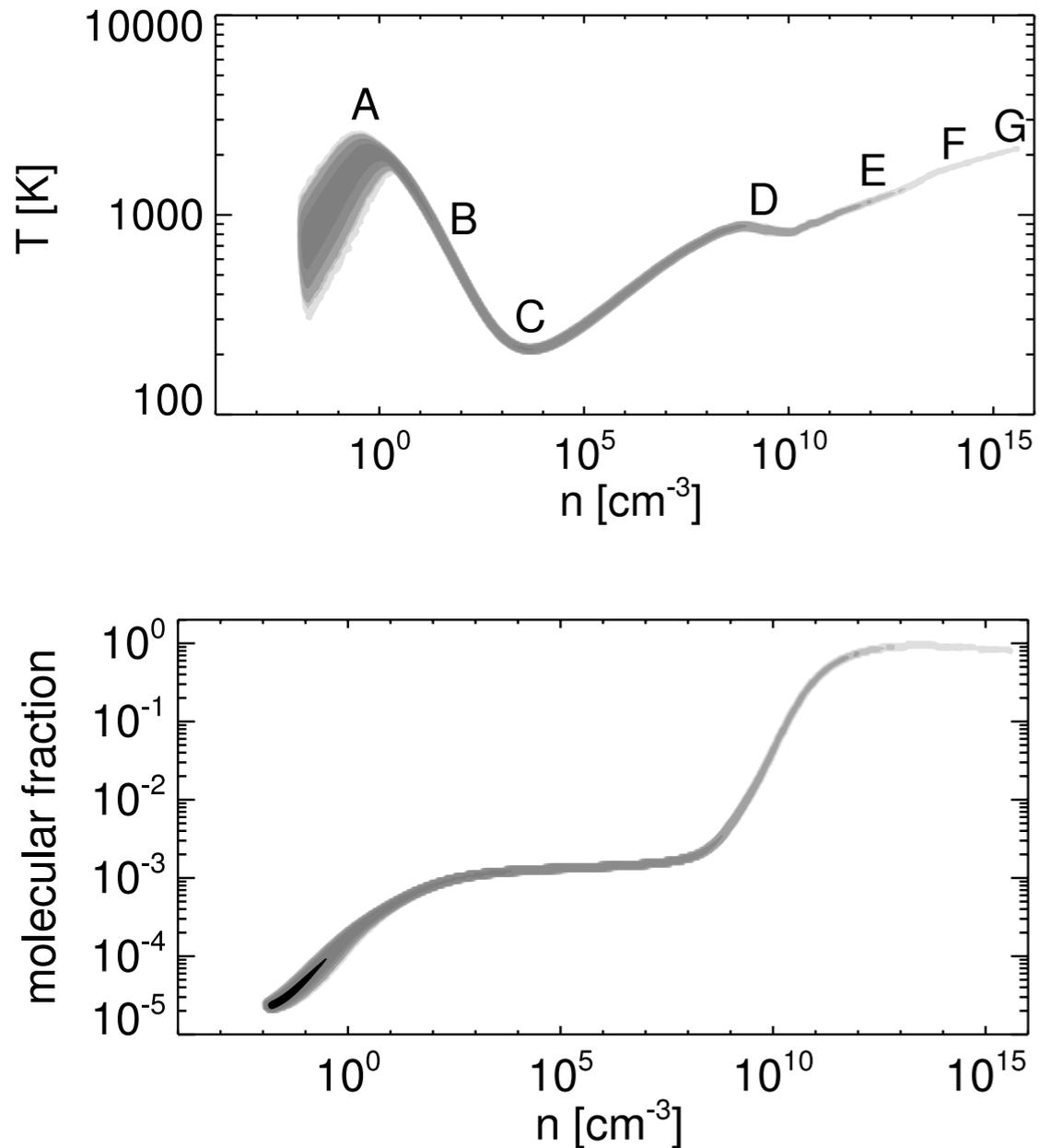
Alternatively:

- Continue your own SPH projects from last week
- Start something new using the features in Seren

Simulating chemically reactive flows in astrophysics

Based on a talk given by Simon Glover
(ZAH/ITA, Heidelberg)
IMPRS Summer School 2012

Why study chemically reactive flows?

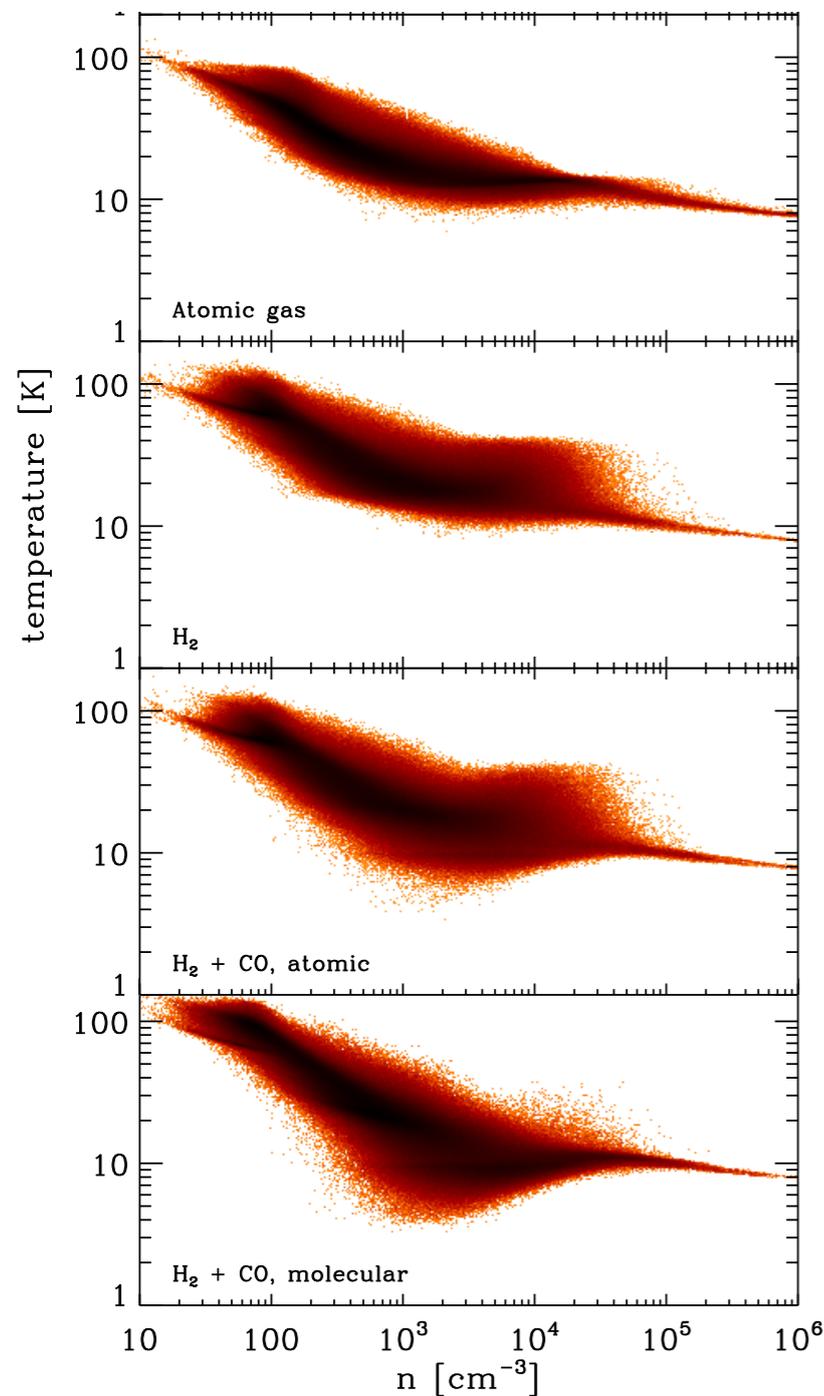


Yoshida et al. (2006)

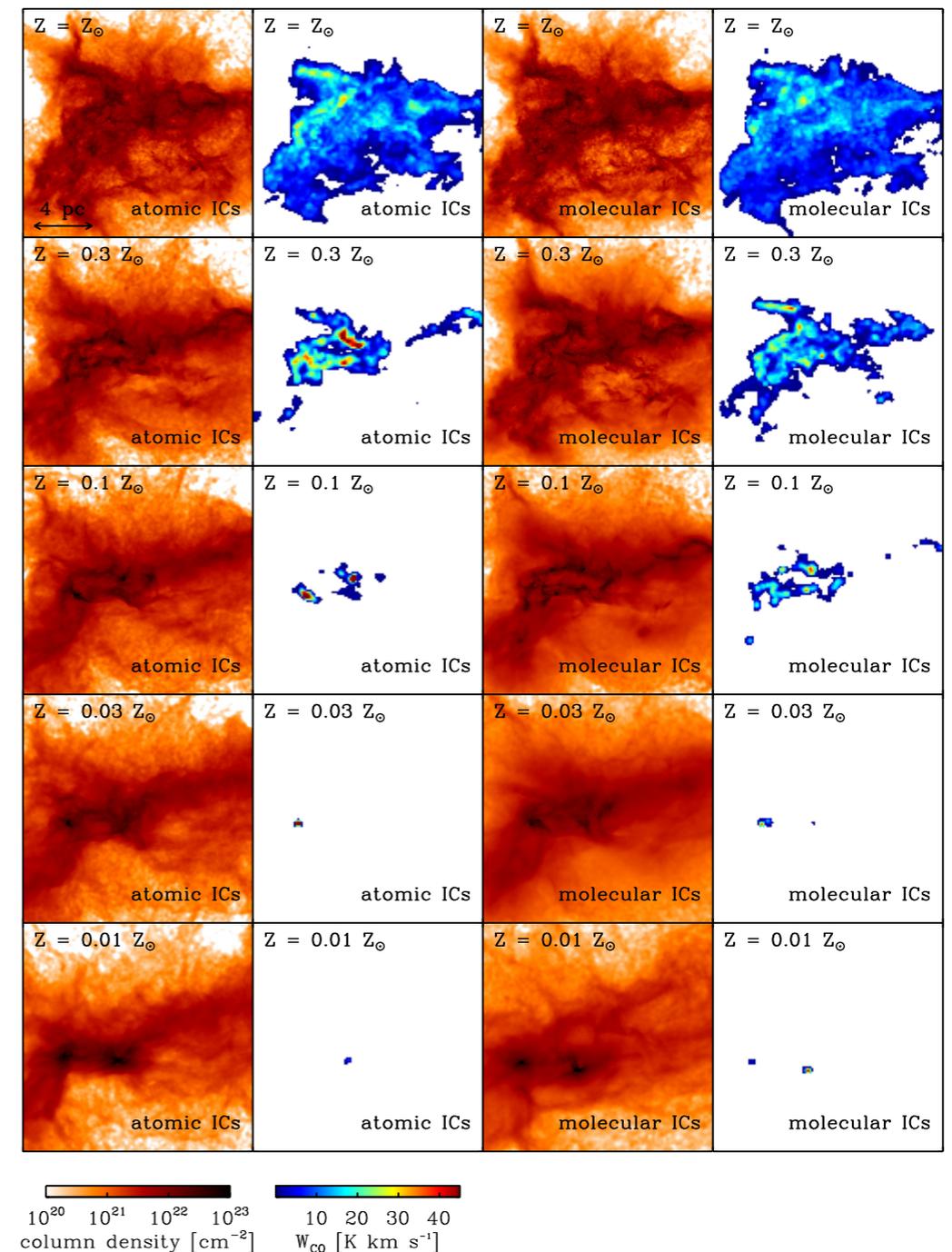
Stacy, Greif and Bromm (2010)

Why study chemically reactive flows?

Glover & Clark (2012a)



Glover & Clark (2012b)



Governing equations

- The number density n_i of a chemical species i evolves according to:

$$\frac{\partial n_i}{\partial t} + \vec{\nabla} \cdot (n_i \vec{v}) - \vec{\nabla} \cdot (\mathcal{D} \vec{\nabla} n_i) = C(n_i, n_j, T) - D(n_j, T) n_i$$

- Three processes are acting to change n_i : advection, diffusion and chemical reactions (RHS)
- Convenient to write the reaction term in two parts, one corresponding to formation (C) and the other to destruction (D)

Governing equations

- The number density n_i of a chemical species i evolves according to:

$$\frac{\partial n_i}{\partial t} + \vec{\nabla} \cdot (n_i \vec{v}) - \vec{\nabla} \cdot (\mathcal{D} \vec{\nabla} n_i) = C(n_i, n_j, T) - D(n_j, T) n_i$$

- Three processes are acting to change n_i : advection, diffusion and chemical reactions (RHS)
- Convenient to write the reaction term in two parts, one corresponding to formation (C) and the other to destruction (D)

Governing equations (II)

- In astrophysical flows, we generally ignore the diffusion term, as it is much smaller than the advection term
- Characteristic diffusion length scale is given by:

$$L_{\text{diff}} \sim (D t)^{1/2}$$

- The diffusion coefficient $D \sim L_{\text{mfp}} v_{\text{th}}$, where L_{mfp} is the particle mean free path and v_{th} is the thermal velocity
- In the diffuse ISM ($n = 1 \text{ cm}^{-3}$, $T = 10^4 \text{ K}$), we find that $L_{\text{diff}} \sim 10^{11} t^{1/2} \text{ cm}$ (about 1 pc in 10 Myr).

Governing equations (III)

- We are therefore left with a set of advection-reaction equations, one for each chemical species
- We generally simplify this set of equations using a technique called **operator splitting**
- Instead of solving the full equations, we separate the advection and reaction portions separately:

$$\frac{\partial n_i}{\partial t} + \vec{\nabla} \cdot (n_i \vec{v}) = 0$$

$$\frac{dn_i}{dt} = C(n_i, n_j, T) - D(n_j, T)n_i$$

Operator splitting

- Operator splitting the advection and reaction steps has a drawback: the introduction of a new source of truncation error
- Simplest splitting strategy:
 - Evolve advection eq. from t_0 to t_1
 - Using output from advection step, evolve reaction network from t_0 to t_1
 - Alternatively, we can invert the order of the steps, and solve the reaction substep first, followed by the advection substep

Operator splitting (II)

- This scheme is called “first-order splitting”. It introduces a local error of $O(\Delta t)$.
- We can do better than this with a scheme known as “Strang splitting”:
 - Evolve reaction network from t_0 to $t_0 + \Delta t/2$
 - Evolve advection step from t_0 to t_1
 - Evolve reaction network from $t_0 + \Delta t/2$ to t_1
- This scheme introduces a local error of $O(\Delta t^2)$.

Operator splitting (III)

- Note that if the reaction network is stiff, then we always want to evolve it last in our splitting scheme.
- This is to ensure that rapidly reacting species that should be in chemical equilibrium are indeed in equilibrium at the end of every timestep

Advection

- In general, we can use the same techniques to advect our set of number densities n_i that we use to advect the mass density ρ
- However, since we must conserve the total quantity of each element (hydrogen, helium, carbon etc.), plus the total charge, our number densities must also satisfy a set of $N_{\text{elem}} + 1$ constraint equations
- In Eulerian codes, the advection scheme typically does not guarantee that these constraint equations remain satisfied

Advection (II)

- We can ensure that the constraint equations are satisfied at the end of every advection step by directly adjusting our number densities n_i
- However, this strategy can be highly diffusive (see e.g. Plewa & Mueller, 1999)
- Plewa & Mueller suggest that one should instead adjust the fluxes of the various species to ensure local conservation

Consistent Multi-species Advection (CMA)

- Total flux of element a with mass fraction x_a :

$$F_a = x_a F_\rho$$

- Partial fluxes F_i do not, in general, sum to total flux F_a

$$F_a \neq \sum_i F_i$$

- Hence have to rescale the fluxes by a factor:

$$\phi = \frac{F_a}{\sum_i F_i}$$

In SPH...

- Advection in SPH is done by moving the particles directly.
- No need to worry about the advection errors (at least in terms of what they are doing to the species conservation).
- Chemical rate equation simply becomes:

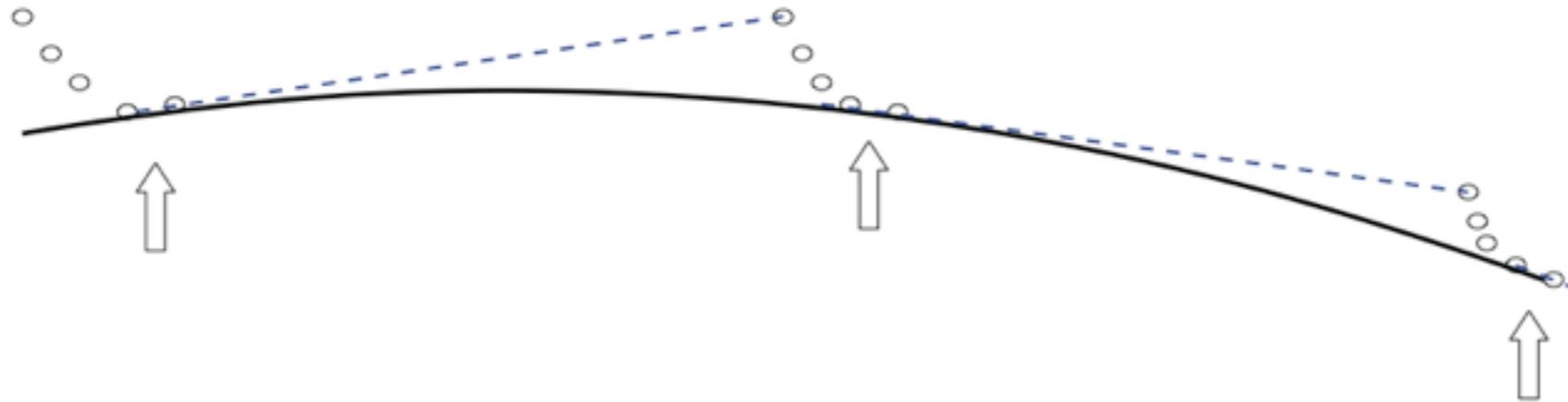
$$\frac{dn_i}{dt} = C - D n_i$$

- If including the diffusion terms (e.g. Greif et al. 2009), then this can in principle become an issue.
- Also need to consider operator splitting the fluxes from the chemistry solve.

Reactions

- A typical chemical reaction network contains processes with a wide range of characteristic timescales
- The resulting rate equations are **stiff**
- This is a big problem: stiff ODEs can be solved explicitly only if very small timesteps are taken

Stiffness and stability



From Lee & Gear, 2007, JACM, 201, 258

- Consider a system with a set of fast modes that rapidly reach equilibrium, and a set of slow modes that control the subsequent evolution
- Away from the equilibrium manifold, we get rapid evolution and explicit codes must take small timesteps
- BUT: errors (truncation & roundoff), and advection effects will always displace us from the equilibrium manifold

Stiffness and stability (II)

- We can avoid numerical instability by using an implicit technique
- One of the simplest possible implicit solvers is the 1st order backwards differencing formula (BDF):

$$\frac{n_i(t_1) - n_i(t_0)}{\Delta t} = C(t_1) - D(t_1)n_i(t_1)$$

$$n_i(t_1) = \frac{n_i(t_0) + C(t_1)\Delta t}{1 + D(t_1)\Delta t}$$

1st order BDF: pros and cons

- Pros:
 - Simple
 - Fast
 - Solutions are certain to remain ≥ 0
- Cons:
 - Inaccurate
 - No error control

Other solvers

Can control the accuracy of the solution:

- Higher order BDF (e.g. LSODE, VODE)
- Implicit Runge-Kutta (e.g. SDIRK)
- Rosenbrock (e.g. RODAS)
- For a good overview of different techniques, see Sandu et al. (1997a,b)

Caution when using high-order solvers

- Discontinuities in the rate equations can produce problems when computing the Jacobian.
- Need to ensure the derivatives are continuous: see work by Marcus Röllig (2007)
- In high order solvers, the solution is not guaranteed to be **positive**.
- Need to sanity-check the output.
- Require the chemical reaction rates to be in a somewhat abstract form -- not ideal for getting to know how something works!

Benefits?

Aside from the accuracy issue:

- Adding new reactions to the set is trivial.
- No need to worry about how stiff the new reaction is compared to the others.
- Automatically “sub-cycles” to the required timestep
- These subroutines are often highly optimised.
- Many are also still maintained (less common with the older F77 routines).

Performance

- Implicit solvers typically require some form of Newton iteration
- Solution of a set of N_{sp} coupled equations requires the inversion of a $N_{sp} \times N_{sp}$ matrix
- Computational cost scales as N_{sp}^3
- Values N_{sp} of = 50 - 400 are not uncommon in astrochemistry

Improving performance (I): exploiting sparseness

- Typically, any given reactant will react with only a small subset of the total number of species
- Hence our $N_{sp} \times N_{sp}$ matrix is sparse
- We can get an easy speedup by using a solver tailored for sparse systems
- Typical speedups of a factor of a few
- See Timmes 1999, Nejad 2005 for examples

Improving performance (II): dimension reduction

- If our cost is scaling as N_{sp}^3 , then we can run faster if we can reduce N_{sp}
- An easy way to do this is by reducing the size of our chemical network
- Analysis of our chemical network can identify reactions and/or reactants that are unimportant and that can safely be omitted
- To perform this analysis, we need to understand the region of n - T - A_V -space in which our models will evolve

Improving performance (II): dimension reduction

- The more we know about our application, the less conservative we need to be when reducing the kinetics
- Nevertheless, we're trading computer time for person time
- It would be convenient if we could automatically reduce the number of dimensions whenever appropriate without requiring human intervention
- This desire prompted the development of a technique called Computational Singular Perturbation (CSP)

CSP

- We can express our N_{sp} rate equations as a vector \mathbf{g} in an N_{sp} -dimensional vector space
- By choosing an appropriate set of basis vectors, we can transform \mathbf{g} so as to decouple rapidly reacting modes from slowly reacting modes
- If we then assume that the rapidly reacting modes are in equilibrium, then we only need to solve for the time evolution of the slowly reacting modes
- We can do this dynamically - i.e. we can recompute our decomposition whenever the fluid properties change
- For full details, see Lam (1993)

- This talk will be posted online (pdf)

<http://www.ita.uni-heidelberg.de/~pcc/hipacc>

A few useful references

General references

- Oran, E. & Boris, J., 2000, “Numerical Simulation of Reactive Flow”, CUP
- Sportisse, B. 2007, Comput. Geosci., 11, 159

Implicit ODE solvers

- Hairer, E., Wanner, G., 1996, “Solving Ordinary Differential Equations, volume II: Stiff and differential algebraic problems”, Springer Verlag
- Timmes, F.X. 1999, ApJS, 124, 241
- Nejad, 2005, Ap&SS, 299, 1
- Sandu, A., et al., 1997a, Atmos. Environ, 31, 3151
- Sandu, A., et al., 1997b, Atmos. Environ, 31, 3459

A few more useful references

Other topics

- Plewa, T. & Müller, E. 1999, A&A, 342, 179 (CMA)
- Lam, 1993, Comb. Sci. Tech., 89, 375 (CSP)
- Pope, S. 1997, Comb. Theo. Modelling, 1, 41 (ISAT)

Seren download guide

Website: <http://dhubber.github.io/seren/seren.html>

```
[pcc@hyades ~]$ git clone https://github.com/dhubber/seren.git
```

If you want to use the chemistry, you need to change to the 'chemistry' branch:

```
[pcc@hyades ~]$ git checkout -b chemistry
```

```
[pcc@hyades ~]$ git pull origin chemistry
```

What's where?

```
[pcc@hyades seren]$ ls
COPYING      docs      joinsims  makefiletail.mk  README.md  seren
datafiles    examples  Makefile  params.dat       scripts    src
```

```
[pcc@hyades seren]$ cd examples/
[pcc@hyades examples]$ ls
bossbodenheimer  sedov  sodshocktube  wallshock
[pcc@hyades examples]$ cd bossbodenheimer/
[pcc@hyades bossbodenheimer]$ ls
204800_BBSIT.sf.dat.gz  3200_BBSIT.sf.dat.gz  Makefile  seren
25600_BBSIT.sf.dat.gz  asciicolumns.dat      params.dat
```

IC files

describes what property is in what column

example parameter file

For chemistry

```
[pcc@hyades seren]$ ls
COPYING      docs      ic_BB      Makefile      params.dat  scripts  src
datafiles    examples  joinsims    makefiletail.mk  README.md  seren
[pcc@hyades seren]$ cd src/
[pcc@hyades src]$ ls
advance      ghosts    io          radial_average.F90  sph
analyse      gravity   main        radiation           sph_sim
BHtree       headers   mpi         setup              tests
chemcool     healpix   nbody       sinks              timestep
devel        ic        nbody_sim   sorts              turbulence
[pcc@hyades src]$ cd chemcool/
[pcc@hyades chemcool]$ ls
chemical_rate_coeff.F90  do_chemcool_step.F90      solve_chem_timestep.F90
chemistry_constants.F90  ism_heat_cool_rate.F90
compute_stim.F90         one_zone_chemcool.F90
```

Interface between seren and chemistry

The iterative solver

To switch the chemistry on...

- In Makefile, need to set:
 - CHEMCOOL = 1
 - NCHEM = 2
- Need to add the abundance to the data files:
 - Add 'abundh2' and 'abundhp' to the file 'asciicolumns.dat'
 - In your initial conditions you will need to set these to the starting abundance.

To switch the chemistry on...

- In Makefile, need to set:
 - CHEMCOOL = 1
 - NCHEM = 2
- Need to add the abundance to the data files:
 - Add 'abundh2' and 'abundhp' to the file 'asciicolumns.dat'
 - In your initial conditions you will need to set these to the starting abundance.