2010 HIPACC Astro-Computing Summer School

Galaxy Simulations Using the N-Body/SPH code GADGET

T.J. Cox (Carnegie Observatories)

Project 1.1: Code Optimization

<u>PBS batchscript :</u>

```
#!/bin/sh
#PBS -q batch
#PBS -N Sbc
#PBS -l nodes=2:ppn=2
#PBS -o Sbc.out
#PBS -e Sbc.err
#PBS -V
#PBS -M tcox@obs.carnegiescience.edu
#PBS -m abe

cd /home/hipacc-5/Sbc/

mpirun -v -machinefile $PBS_NODEFILE -np 4 ./Gadget2 Sbc.txt > output0.txt
```

While you're free to email me, your batch scripts should really contain <u>your</u> email address.

Outline

1. ~~Who am I and what am I doing here? My perspective, my science, and where my focus will be this week~~
2. ~~An overview of GADGET projects (+other practical - I hope - information)~~
3. A brief overview of GADGET
4. Adding "Astrophysics" to GADGET
5. Loose Ends ... data structures, analysis, and visualization (w/ P. Hopkins)
6. What's next? (higher resolution, new models, and Arepo: the next generation of code)

3. GADGET: A Brief Intro

3.1 The Monte Carlo, N-body approach to solving the CBE

3.2 Gravity calculation

3.3 Integration and time-steps

3.4 Including hydrodynamics with SPH

3.5 The steps Gadget takes to accomplish the above

3.7 Data Structures within Gadget

3.8 Modes of Gadget

3.9 The remaining compile-time and run-time parameters

3.10 Odd and Ends ... Questions

Gadget (and other N-body) Resources

- Gadget Manual - comes with the public download
- Gadget papers: 1 (Springel, Yoshida, & White 2000) and 2 (Springel 2005)
- Hernquist & Katz (1989): TreeSPH (basically, Gadget version 0)
- Josh Barnes: (Barnes & Hut 1985, Barnes' website, his 1996 Saas-Fee lectures)
- Volker Spingel's 2009 IAS Summer School lectures (very technical, but a thorough introduction to the nitty-gritty)
- Binney & Tremaine, *Galactic Dynamics* (1987)
- Hands-on experience digging within the code, modifying it, screwing it up and trying to figure out how to fix it again ....

3.1 Mont Carlo Approach to solving the Collisionless Boltzmann Equation

Galaxies are collisionless systems; t_relax ~ (N/8LnLambda) t_cross (see Binney & Tremaine for a nice discussion of this)

BASIC MONTE-CARLO IDEA

Collisionless Boltzmann equation

Poisson-Vlasov System

$$\frac{\mathrm{d}f}{\mathrm{d}t} = \frac{\partial f}{\partial t} + \frac{\partial f}{\partial \mathbf{x}} \cdot \mathbf{v} + \frac{\partial f}{\partial \mathbf{v}} \cdot \left(-\frac{\partial \Phi}{\partial \mathbf{x}}\right) = 0$$

$$\nabla^2 \Phi(\mathbf{x}, t) = 4\pi G \int f(\mathbf{x}, \mathbf{v}, t)\, \mathrm{d}\mathbf{v}$$

N-body System

$$\ddot{\mathbf{x}}_i = -\nabla_i \Phi(\mathbf{x}_i)$$

$$\Phi(\mathbf{x}) = -G \sum_{j=1}^{N} \frac{m_j}{[(\mathbf{x} - \mathbf{x}_j)^2 + \epsilon^2]^{1/2}}$$

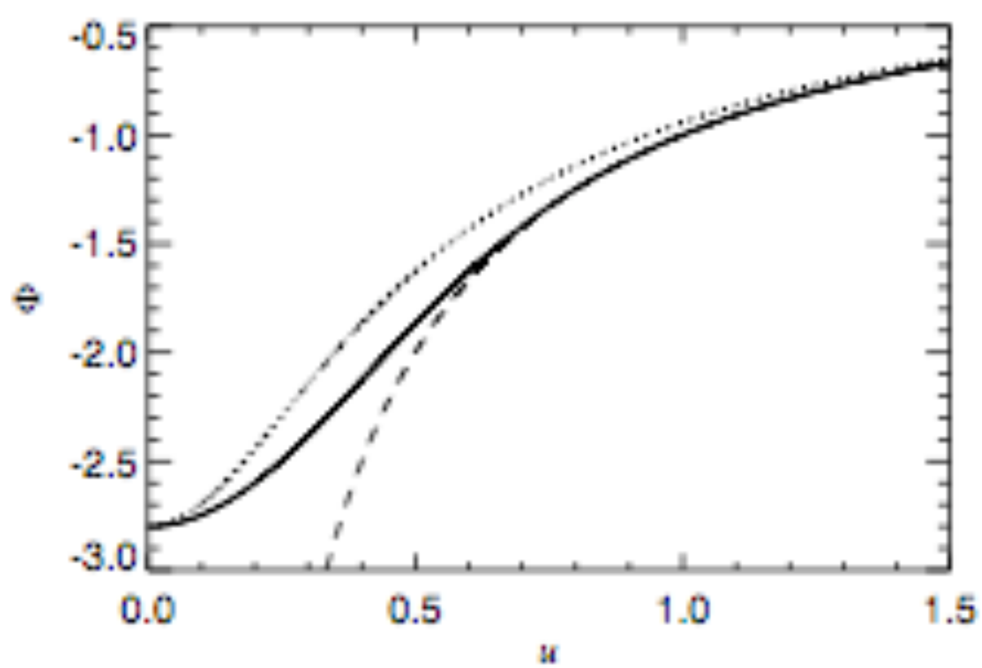
need large **N**

3.1 Mont Carlo Approach to solving the Collisionless Boltzmann Equation

N-body System

$$\ddot{\mathbf{x}}_i = -\nabla_i \, \Phi(\mathbf{x}_i)$$

$$\Phi(\mathbf{x}) = -G \sum_{j=1}^{N} \frac{m_j}{[(\mathbf{x} - \mathbf{x}_j)^2 + \epsilon^2]^{1/2}}$$



Gravitational softening within some scale accounts for finite N. Gadget uses spline kernel.

3.1 Mont Carlo Approach to solving the Collisionless Boltzmann Equation

What should the gravitational softening be? .....
unfortunately, this isn't an easy question to answer.

* To zeroth order, collisionless criterion suggest it should depend on N and t_cross
* Cosmological simulations often employ simple criterion based upon the mean inter-particle spacing (~1/20th or so)
* Power et al. (2003) present a nice discussion and argue for values based on N and the size of the DM halo

What should the gravitational softening be? .....
unfortunately, this isn't an easy question to answer.

* To zeroth order, collisionless criterion suggest it should depend on N and t_cross
* Cosmological simulations often employ simple criterion based upon the mean inter-particle spacing (~1/20th or so)
* Power et al. (2003) present a nice discussion and argue for values based on N and the size of the DM halo

While these estimates are useful starting points, there is NO definitive way to know what the softening should be outside of performing detailed numerical experiments.

3.1 Mont Carlo Approach to solving the Collisionless Boltzmann Equation

What should the gravitational softening be? .....
unfortunately, this isn't an easy question to answer.

Compile-time options from within the Makefile:

```
OPT    +=  -DUNEQUALSOFTENINGS
```

Parameter file options:

```
% Softening lengths

MinGasHsmlFractional 0.25

SofteningGas        0.1
SofteningHalo       0.2
SofteningDisk       0.1
SofteningBulge      0.1
SofteningStars      0.1
SofteningBndry      0.1

SofteningGasMaxPhys       0.1
SofteningHaloMaxPhys      0.2
SofteningDiskMaxPhys      0.4
SofteningBulgeMaxPhys     0.1
SofteningStarsMaxPhys     0.1
SofteningBndryMaxPhys     0.1
```

3.1 Mont Carlo Approach to solving the Collisionless Boltzmann Equation

What should the number of particles, N, be?

3.1 Mont Carlo Approach to solving the Collisionless Boltzmann Equation

What should the number of particles, N, be?

* The answer to this question is easy - as large as possible!

What should the number of particles, N, be?

* The answer to this question is easy - as large as possible!

Of course, N_simulated will always (at least for the near future) be much smaller than N_actual, so the only way to know what affect the choice of N plays on your results is to perform detailed numerical experiments.

3.1 Mont Carlo Approach to solving the Collisionless Boltzmann Equation

What should the number of particles, N, be?

An example:

A current "typical" run (moderately high resolution, but quick enough to run numerous runs) has 1 million DM particles, 300k gas and collisionless disk particles, and 100k bulge particles. The gravitational softening was 70 pc for all baryonic components and 250 pc for the DM.

-> isolated galaxy evolved for 3 Gyr: 5 days on 8 processors

-> major merger between two of these: 45 days on 32 processors

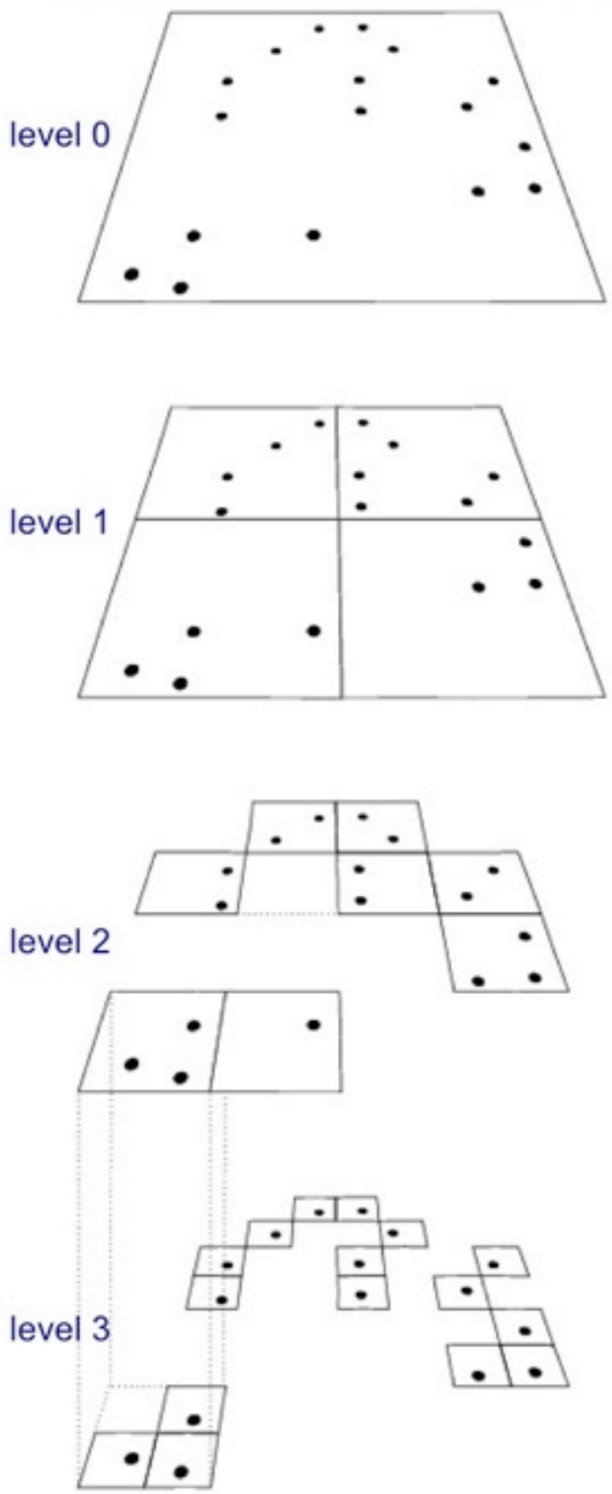
Highest resolution mergers:

25 million total / 1 million gas - ~4 months to completion (128 processors)

15 million total / 3 million gas - still running after 6 months (48 processors)

3.3 Gravity calculation

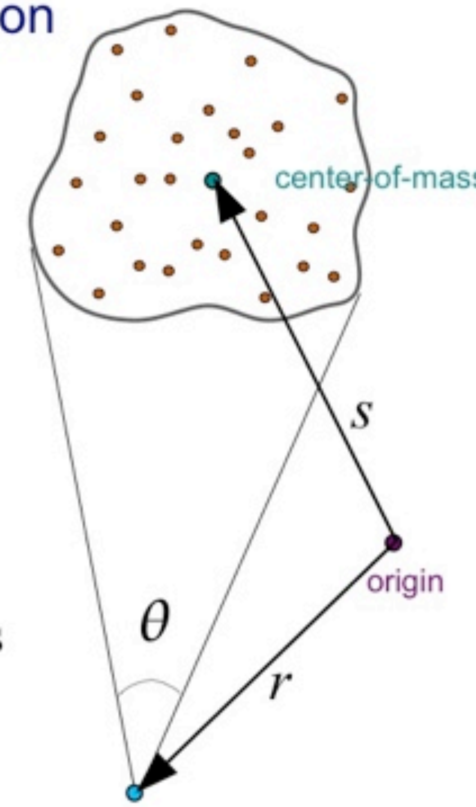
Oct-tree in two dimensions



Tree algorithms

Idea: Use hierarchical multipole expansion to account for distant particle groups

$$\Phi(\mathbf{r}) = -G \sum_i \frac{m_i}{|\mathbf{r} - \mathbf{x}_i|}$$



We expand:

$$\frac{1}{|\mathbf{r} - \mathbf{x}_i|} = \frac{1}{|(\mathbf{r} - \mathbf{s}) - (\mathbf{x}_i - \mathbf{s})|}$$

for $|\mathbf{x}_i - \mathbf{s}| \ll |\mathbf{r} - \mathbf{s}|$ $\quad \mathbf{y} \equiv \mathbf{r} - \mathbf{s}$

and obtain:

$$\frac{1}{|\mathbf{y} + \mathbf{s} - \mathbf{x}_i|} = \frac{1}{|\mathbf{y}|} - \frac{\mathbf{y} \cdot (\mathbf{s} - \mathbf{x}_i)}{|\mathbf{y}|^3} + \frac{1}{2} \frac{\mathbf{y}^T \left[3(\mathbf{s} - \mathbf{x}_i)(\mathbf{s} - \mathbf{x}_i)^T - \mathbf{I}(\mathbf{s} - \mathbf{x}_i)^2 \right] \mathbf{y}}{|\mathbf{y}|^5} + \ldots$$

the dipole term vanishes when summed over all particles in the group

The multipole moments are computed for each node of the tree

Monpole moment:

$$M = \sum_i m_i$$

Quadrupole tensor:

$$Q_{ij} = \sum_k m_k \left[3(\mathbf{x}_k - \mathbf{s})_i (\mathbf{x}_k - \mathbf{s})_j - \delta_{ij} (\mathbf{x}_k - \mathbf{s})^2 \right]$$

Resulting potential/force approximation:

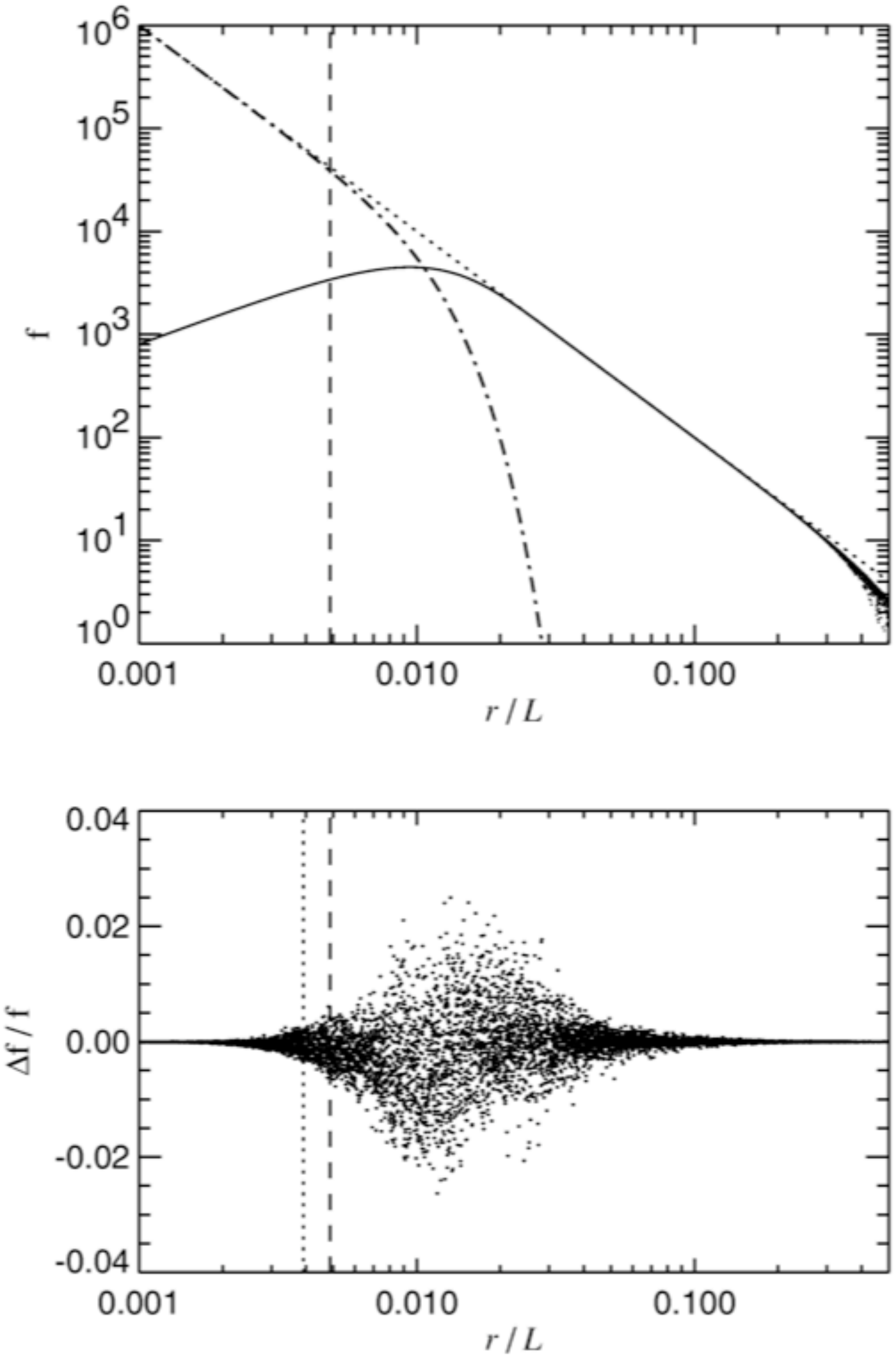
$$\Phi(\mathbf{r}) = -G \left[\frac{M}{|\mathbf{y}|} + \frac{1}{2} \frac{\mathbf{y}^T \mathbf{Q}\, \mathbf{y}}{|\mathbf{y}|^5} \right]$$

For a single force evaluation, not *N* single-particle forces need to be computed, but **only of order *log(N)* multipoles**, depending on the opening angle.

- The tree algorithm has no intrinsic restrictions for its dynamic range
- force accuracy can be conveniently adjusted to desired level
- the speed does depend only very weakly on clustering state
- geometrically flexible, allowing arbitrary geometries

3.3 Gravity calculation

Gadget2 can also calculate gravitational forces via the TreePM method.



3.3 Gravity calculation

Compile-time options from within the Makefile:

```
#--------------------------------------- TreePM Options
#OPT   +=  -DPMGRID=128
#OPT   +=  -DPLACEHIGHRESREGION=3
#OPT   +=  -DENLARGEREGION=1.2
#OPT   +=  -DASMTH=1.25
#OPT   +=  -DRCUT=4.5
```

Parameter file options:

```
% Tree algorithm, force accuracy, domain update frequency

ErrTolTheta               0.5
TypeOfOpeningCriterion 1
ErrTolForceAcc            0.005


TreeDomainUpdateFrequency    0.1
```

Leapfrog integrator (requires a single force computation, conserves phase space/symplectic, and is time reversible)

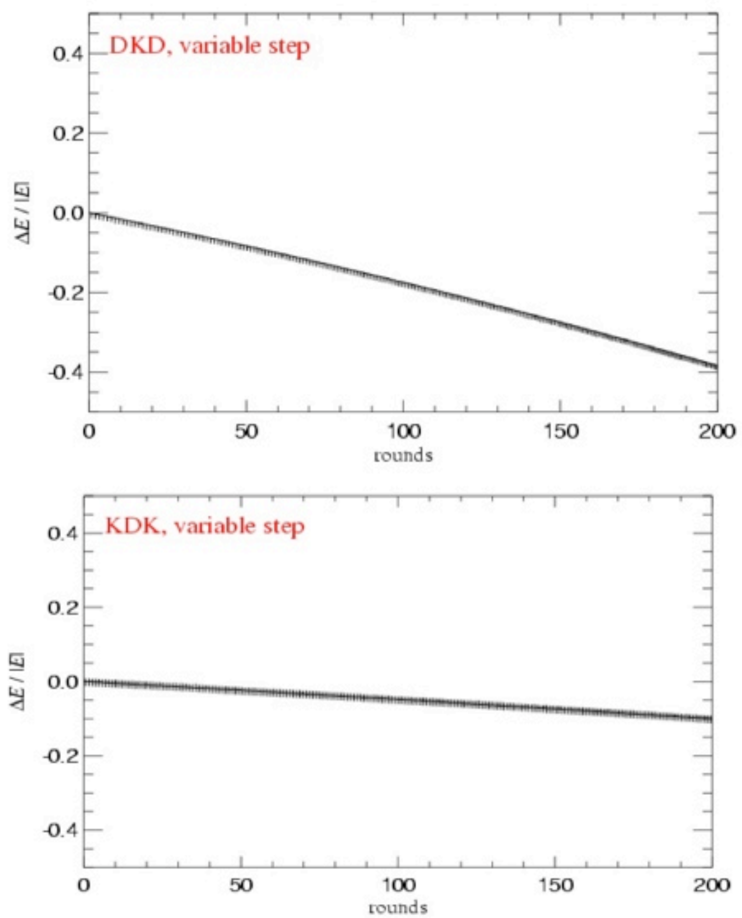
$$
\begin{aligned}
\mathbf{r}_{n+1/2} &= \mathbf{r}_n + \tfrac{1}{2}\tau \mathbf{v}_n, && \text{drift} \\
\mathbf{v}_{n+1} &= \mathbf{v}_n + \tau \mathbf{a}(\mathbf{r}_{n+1/2}), && \text{kick} \\
\mathbf{r}_{n+1} &= \mathbf{r}_{n+1/2} + \tfrac{1}{2}\tau \mathbf{v}_{n+1}, && \text{drift}
\end{aligned}
$$

see Quinn et al. (1997) for much more detail and a much better description

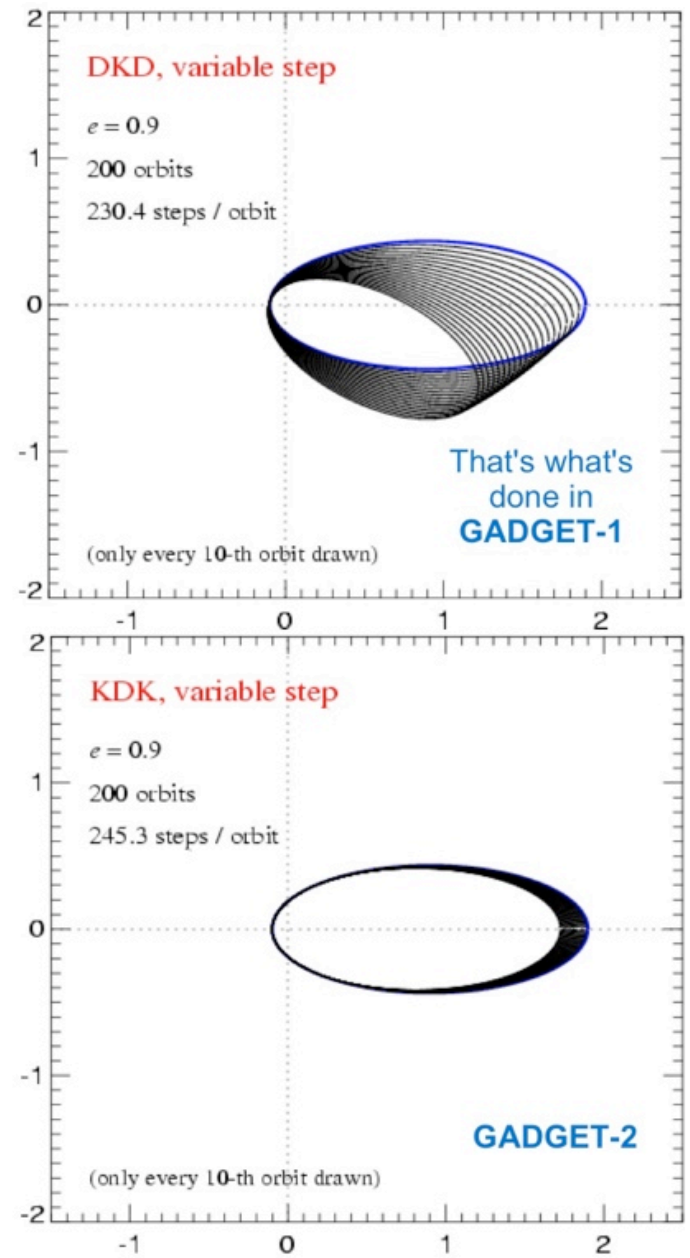
3.2 Integration and time-step issues

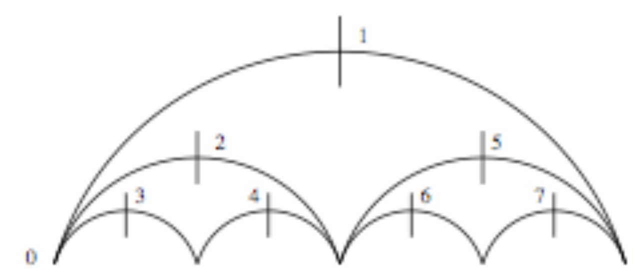
When an adaptive timestep is used, much of the symplectic advantage is lost

INTEGRATING THE KEPLER PROBLEM



→ **Going to KDK reduces the error by a factor 4, at the same cost !**





the situation can improved with timesteps chosen in a factor of 2 hierarchy, see, e.g., Quinn et al. (1997)

Compile-time options from within the Makefile:

```
#--------------------------------------- Time integration options
OPT    +=  -DSYNCHRONIZATION
#OPT    +=  -DFLEXSTEPS
#OPT    +=  -DPSEUDOSYMMETRIC
#OPT    +=  -DNOSTOP_WHEN_BELOW_MINTIMESTEP
#OPT    +=  -DNOPMSTEPADJUSTMENT
```

Parameter file options:

```
% Accuracy of time integration

ErrTolIntAccuracy       0.0025

CourantFac              0.15

MaxSizeTimestep        0.01
MinSizeTimestep        0.0
```

$$\Delta t_{\rm grav} = \min\left[\Delta t_{\rm max}, \left(\frac{2\,\eta\,\epsilon}{|\boldsymbol{a}|}\right)^{1/2}\right]$$

$$\Delta t_i^{\rm (hyd)} = \frac{C_{\rm courant}\, h_i}{\max_j (c_i + c_j - 3w_{ij})}$$

Compile-time options from within the Makefile:

```
#--------------------------------------- Time integration options
OPT   +=  -DSYNCHRONIZATION
#OPT   +=  -DFLEXSTEPS
#OPT   +=  -DPSEUDOSYMMETRIC
#OPT   +=  -DNOSTOP_WHEN_BELOW_MINTIMESTEP
#OPT   +=  -DNOPMSTEPADJUSTMENT
```

Parameter file options:

```
% Accuracy of time integration

ErrTolIntAccuracy       0.0025

CourantFac              0.15

MaxSizeTimestep        0.01
MinSizeTimestep        0.0

TypeOfTimestepCriterion  0
```

$$\Delta t_{\rm grav} = \min\left[\Delta t_{\max},\, \left(\frac{2\,\eta\,\epsilon}{|\boldsymbol{a}|}\right)^{1/2}\right]$$

$$\Delta t_i^{\rm (hyd)} = \frac{C_{\rm courant}\, h_i}{\max_j(c_i + c_j - 3w_{ij})}$$

obsolete in Gadget 2

3.2 Integration and time-step issues

Compile-time options from within the Makefile:

```
#--------------------------------------- Time integration options
OPT    +=  -DSYNCHRONIZATION
#OPT    +=  -DFLEXSTEPS
#OPT    +=  -DPSEUDOSYMMETRIC
#OPT    +=  -DNOSTOP_WHEN_BELOW_MINTIMESTEP
#OPT    +=  -DNOPMSTEPADJUSTMENT
```

Parameter file options:

```
% Accuracy of time integration

ErrTolIntAccuracy        0.0025

CourantFac               0.15

MaxSizeTimestep         0.01
MinSizeTimestep         0.0
```

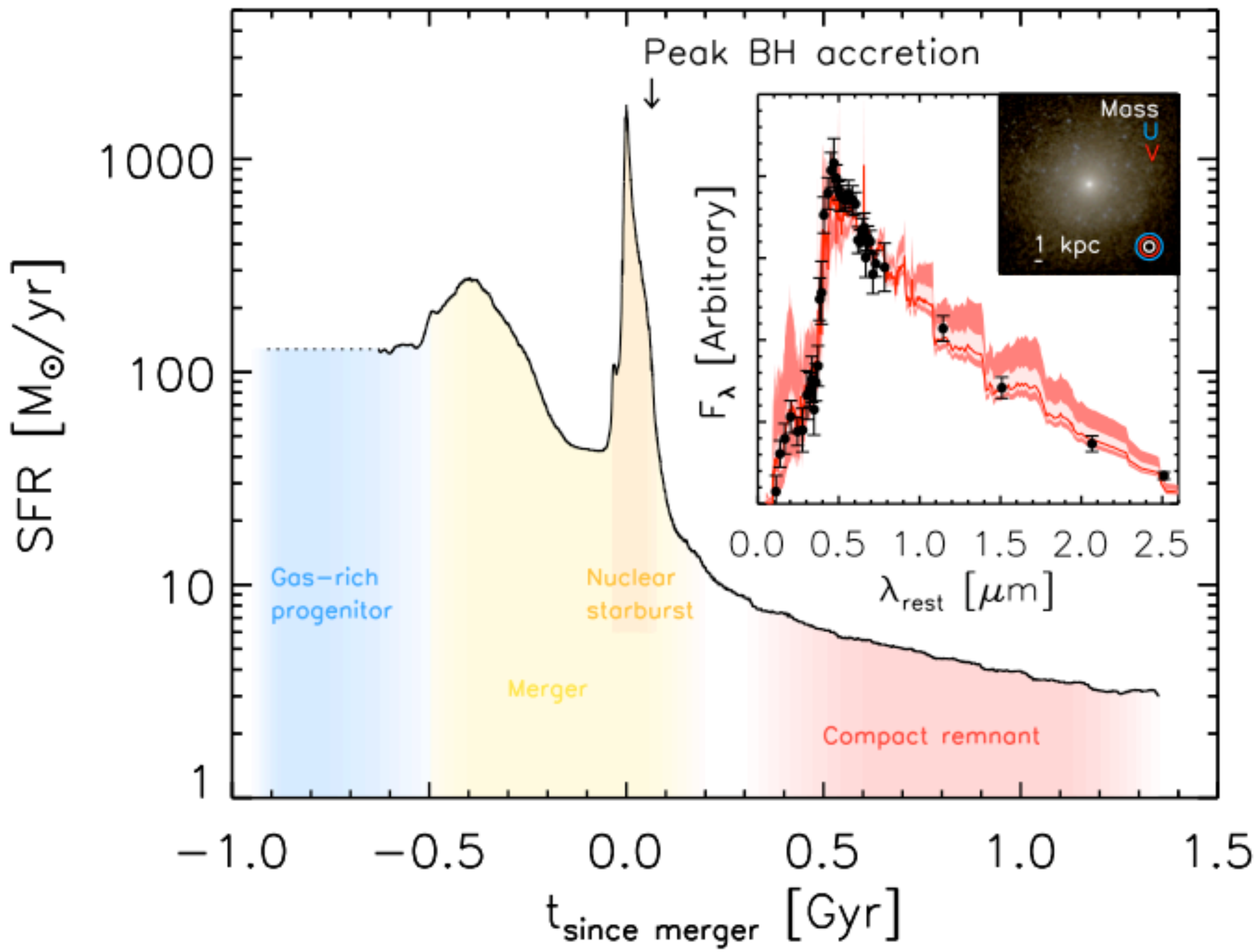
Be Careful with this default value!

$$\Delta t_{\rm grav} = \min\left[\Delta t_{\rm max}, \left(\frac{2\,\eta\,\epsilon}{|\boldsymbol{a}|}\right)^{1/2}\right]$$

$$\Delta t_i^{\rm (hyd)} = \frac{C_{\rm courant}\, h_i}{\max_j (c_i + c_j - 3 w_{ij})}$$

cautionary tale:

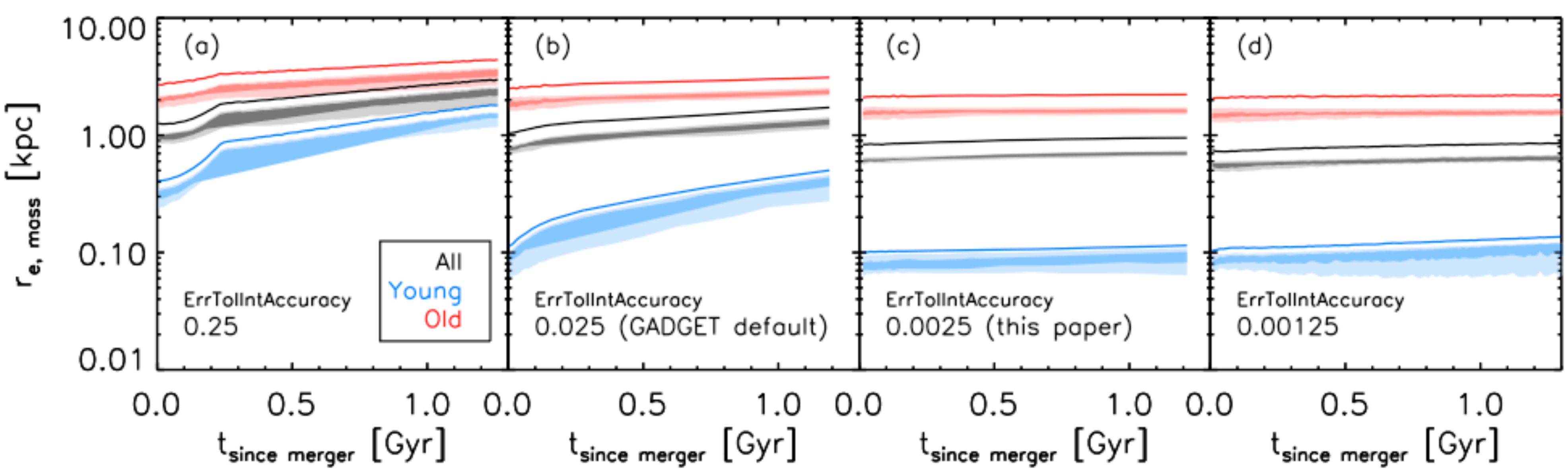
Wuyts et al. (2010) have studies the compact remnants forms from z~2 gas-rich mergers ....



3.2 Integration and time-step issues

cautionary tale:

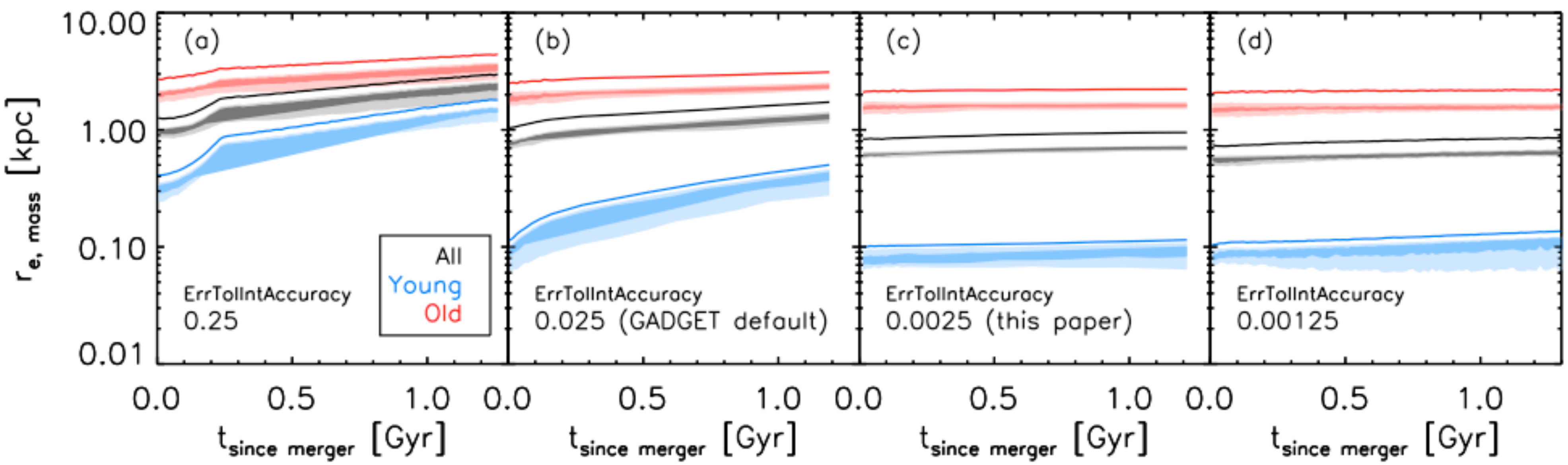
Wuyts et al. (2010) have studies the compact remnants forms from z~2 gas-rich mergers ....



3.2 Integration and time-step issues

cautionary tale:

Wuyts et al. (2010) have studies the compact remnants forms from z~2 gas-rich mergers ....



When looking at structure at/near your resolution limit, you need high integration accuracy!

* Nice discussion from Tom Abel yesterday about various perils of SPH and the new formulation rpSPH

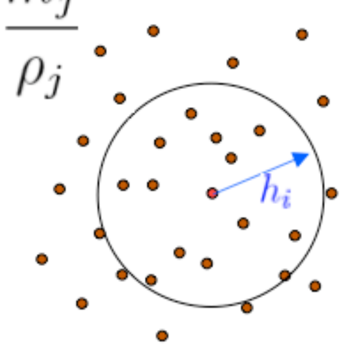
* And from Tom Quinn this morning ..... so this will be brief.

Kernel interpolant of an arbitrary function:

$$\langle A(\mathbf{r}) \rangle = \int W(\mathbf{r} - \mathbf{r}', h)\, A(\mathbf{r}')\, \mathrm{d}^3 r'$$

If the function is only known at a set of discrete points, we approximate the integral as a sum, using the replacement:

$$\mathrm{d}^3 r' \mapsto \frac{m_j}{\rho_j}$$

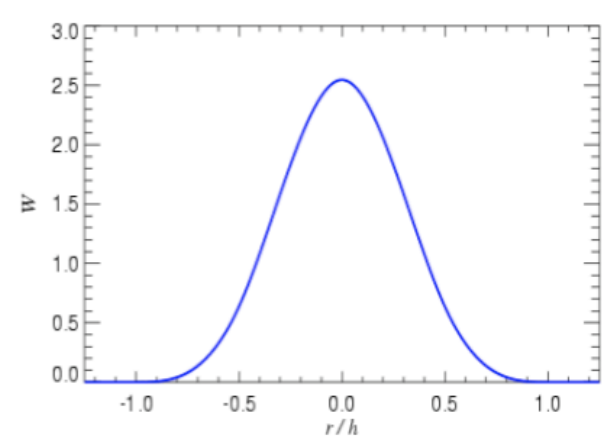


$$\langle A_i \rangle = \sum_{j=1}^{N} \frac{m_j}{\rho_j} A_j\, W(\mathbf{r}_{ij}; h_i)$$

This leads to the SPH density estimate, for $A_i = \rho_i$

$$\rho_i = \sum_{j=1}^{N} m_j W(|\mathbf{r}_{ij}|, h_i)$$

$$W(u) = \frac{8}{\pi} \begin{cases} 1 - 6u^2 + 6u^3, & 0 \le u \le \frac{1}{2}, \\ 2(1-u)^3, & \frac{1}{2} < u \le 1, \\ 0, & u > 1. \end{cases}$$



Smoothed estimate for the velocity field:

$$\langle \mathbf{v}_i \rangle = \sum_j \frac{m_j}{\rho_j} \mathbf{v}_j W(\mathbf{r}_i - \mathbf{r}_j)$$

Velocity divergence can now be readily estimated:

$$\nabla \cdot \mathbf{v} = \nabla \cdot \langle \mathbf{v}_i \rangle = \sum_j \frac{m_j}{\rho_j} \mathbf{v}_j \nabla_i W(\mathbf{r}_i - \mathbf{r}_j)$$

But alternative (and better) estimates are possible also:

Invoking the identity

$$\rho \nabla \cdot \mathbf{v} = \nabla \cdot (\rho \mathbf{v}) - \mathbf{v} \cdot \nabla \rho$$

one gets a “pair-wise” formula:

$$\rho_i (\nabla \cdot \mathbf{v})_i = \sum_j m_j (\mathbf{v}_j - \mathbf{v}_i) \nabla_i W(\mathbf{r}_i - \mathbf{r}_j)$$

BASIC HYDRODYNAMICAL EQUATIONS

Euler equation:	$\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t} = -\frac{\nabla P}{\rho} - \nabla\Phi$
Continuity equation:	$\frac{\mathrm{d}\rho}{\mathrm{d}t} + \rho\nabla\cdot\mathbf{v} = 0$
First law of thermodynamics:	$\frac{\mathrm{d}u}{\mathrm{d}t} = -\frac{P}{\rho}\nabla\cdot\mathbf{v} - \frac{\Lambda(u,\rho)}{\rho}$
Equation of state of an ideal monoatomic gas:	$P = (\gamma - 1)\rho u \,, \qquad \gamma = 5/3$

$$\frac{\mathrm{d}\mathbf{v}_i}{\mathrm{d}t} = -\sum_{j=1}^{N} m_j \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2}\right) \nabla_i \overline{W}_{ij}$$

$+\Pi_{ij}$ Artificial viscosi

automatic

$$\frac{\mathrm{d}u_i}{\mathrm{d}t} = \frac{1}{2}\sum_{j=1}^{N} m_j \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2}\right) \mathbf{v}_{ij}\cdot\nabla_i \overline{W}_{ij}$$

$+\Pi_{ij}$

$$P_i = (\gamma - 1)\rho_i u_i$$

3.4 Including hydrodynamics via SPH

As both Tom Abel and Tom Quinn have mentioned, there are many formulations of SPH, specifically how you symmetrize the Kernel or the pressure terms.

Gadget uses an entropy formulation derived with a variational approach that alleviates some of the problems associated with varying smoothing lengths (see Springel & Hernquist 2002).

Parameter file options:

```
%  Further parameters of SPH

DesNumNgb              50
MaxNumNgbDeviation     2
ArtBulkViscConst       0.8
InitGasTemp            0
MinGasTemp             0
MinGasHsmlFractional 0.25
```

3.6 The steps GADGET takes to accomplish the above

A brief outline of the modules that Gadget uses to perform the aforementioned processes:

main.c → begrun.c (read parameter file, opens all output files, sets the system of units, and calls init.c which reads the initial conditions file, which is actually performed in read_ic.c, and sets all the structures within Gadget)

main.c → run.c
(the primary workhorse of Gadget)

- drifts particles → (saves snapshot first, then moves particles in predict.c)
- writes log information
- updates tree (where needed) → (domain.c)
- computes accelerations → (accel.c)
- kicks particles → (timestep.c)

3.6 The steps GADGET takes to accomplish the above

A brief outline of the modules that Gadget uses to perform the aforementioned processes:

main.c → begrun.c (read parameter file, opens all output files, sets the system of units, and calls init.c which reads the initial conditions file, which is actually performed in read_ic.c, and sets all the structures within Gadget)

main.c → run.c (the primary workhorse of Gadget)

- drifts particles → (saves snapshot first, then moves particles in predict.c)
- writes log information
- updates tree (where needed) → (domain.c)
- computes accelerations → (accel.c)
- kicks particles → (timestep.c)

3.6 The steps GADGET takes to accomplish the above

computes accelerations

(accel.c)

- compute gravitational acceleration in gravtree.c
- determine SPH density in density.c
- compute hydrodynamic forces in hydra.c

3.6 The steps GADGET takes to accomplish the above

computes accelerations

(accel.c)

- compute gravitational acceleration in gravtree.c
- determine SPH density in density.c
- compute hydrodynamic forces in hydra.c

It is fairly straightforward to add a fixed potential to a Gadget simulation:

* gravtree.c/gravity_tree walks tree to compute the gravitational acceleration for each particle and stores this in P[i].GravAccel[j]

* Loop through the particle a second time and add any additional acceleration you desire.

3.6 The steps GADGET takes to accomplish the above

computes accelerations

(accel.c)

- compute gravitational acceleration in gravtree.c
- determine SPH density in density.c
- compute hydrodynamic forces in hydra.c
- } other forces or energies can be included (cooling, star formation feedback, etc.)

3.7 Data Structures within Gadget

There are three primary data structures within Gadget

* All.(xx) = global variables stored on ALL processors

* P[i].Pos[j],Vel[j], ... = particle information, unique to each processor

* SphP[i].Entropy,Pressure, ...= SPH particle information, unique to each processor

=> see allvars.h for a complete listing of all structure variables

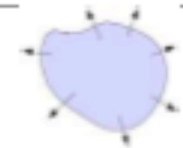
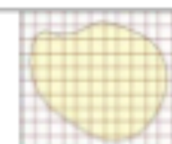
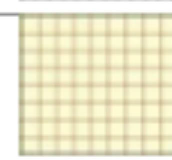
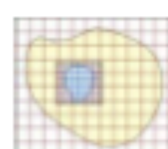
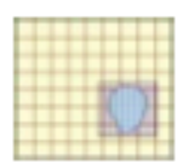
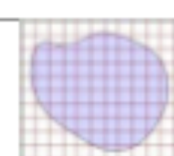
Makefile and parameter settings that need to be changed:

Compile-time options from within the Makefile:

```
#OPT    +=   -DPERIODIC
```

Parameter file options:

```
ComovingIntegrationOn     0
PeriodicBoundariesOn      0
Omega0                    0
OmegaLambda               0
OmegaBaryon               0
HubbleParam             1.0
BoxSize                   0
```

	Type of Simulation		Computational methods	Remarks
1	**Newtonian space**		Gravity: Tree, SPH (optional), vacuum boundary conditions	OmegaLambda should be set to zero
2	**Periodic long box**		No gravity, only SPH, periodic boundary conditions	NOGRAVITY needs to be set, LONG_X/Y/Z may be set to scale the dimensions of the box
3	**Cosmological, physical coordinates**		Gravity: Tree, SPH, vacuum boundaries	ComovingIntegrationOn set to zero
4	**Cosmological, comoving coordinates**		Gravity: Tree, SPH, vacuum boundaries	ComovingIntegrationOn set to one
5	**Cosmological, comoving periodic box**		Gravity: Tree with Ewald-correction, SPH, periodic boundaries	PERIODIC needs to be set
6	**Cosmological, comoving coordinates, TreePM**		Gravity: Tree with long range PM, SPH, vacuum boundaries	PMGRID needs to be set
7	**Cosmological, comoving periodic box, TreePM**		Gravity: Tree with long range PM, SPH, periodic boundaries	PERIODIC and PMGRID need to be set
8	**Cosmological, comoving coordinates, TreePM, Zoom**		Gravity: Tree with long-range and intermediate-range PM, SPH, vacuum boundaries	PMGRID and PLACEHIGHRESREGION need to be set
9	**Cosmological, periodic comoving box, TreePM, Zoom**		Gravity: Tree with long-range and intermediate-range PM, SPH, periodic boundaries	PERIODIC, PMGRID and PLACEHIGHRESREGION need to be set
10	**Newtonian space, TreePM**		Gravity: Tree with long-range PM, SPH, vacuum boundaries	PMGRID needs to be set

3.9 The remaining parameters

The remainder of Gadget compile-time and parameter file options .....

Compile-time options from within the Makefile:

```
OPT    +=  -DPEANOHILBERT

OPT    +=  -DWALLCLOCK

OPT    +=  -DDOUBLEPRECISION
#OPT   +=  -DDOUBLEPRECISION_FFTW


#OPT   +=  -DHAVE_HDF5
OPT    +=  -DOUTPUTPOTENTIAL
#OPT   +=  -DOUTPUTACCELERATION
#OPT   +=  -DOUTPUTCHANGEOFENTROPY
#OPT   +=  -DOUTPUTTIMESTEP
```

3.9 The remaining parameters

The remainder of Gadget compile-time and parameter file options .....

Compile-time options from within the Makefile:

```
OPT    +=  -DPEANOHILBERT

OPT    +=  -DWALLCLOCK

OPT    +=  -DDOUBLEPRECISION
#OPT   +=  -DDOUBLEPRECISION_FFTW

#OPT   +=  -DHAVE_HDF5
OPT    +=  -DOUTPUTPOTENTIAL
#OPT   +=  -DOUTPUTACCELERATION
#OPT   +=  -DOUTPUTCHANGEOFENTROPY
#OPT   +=  -DOUTPUTTIMESTEP
```

always recommended

3.9 The remaining parameters

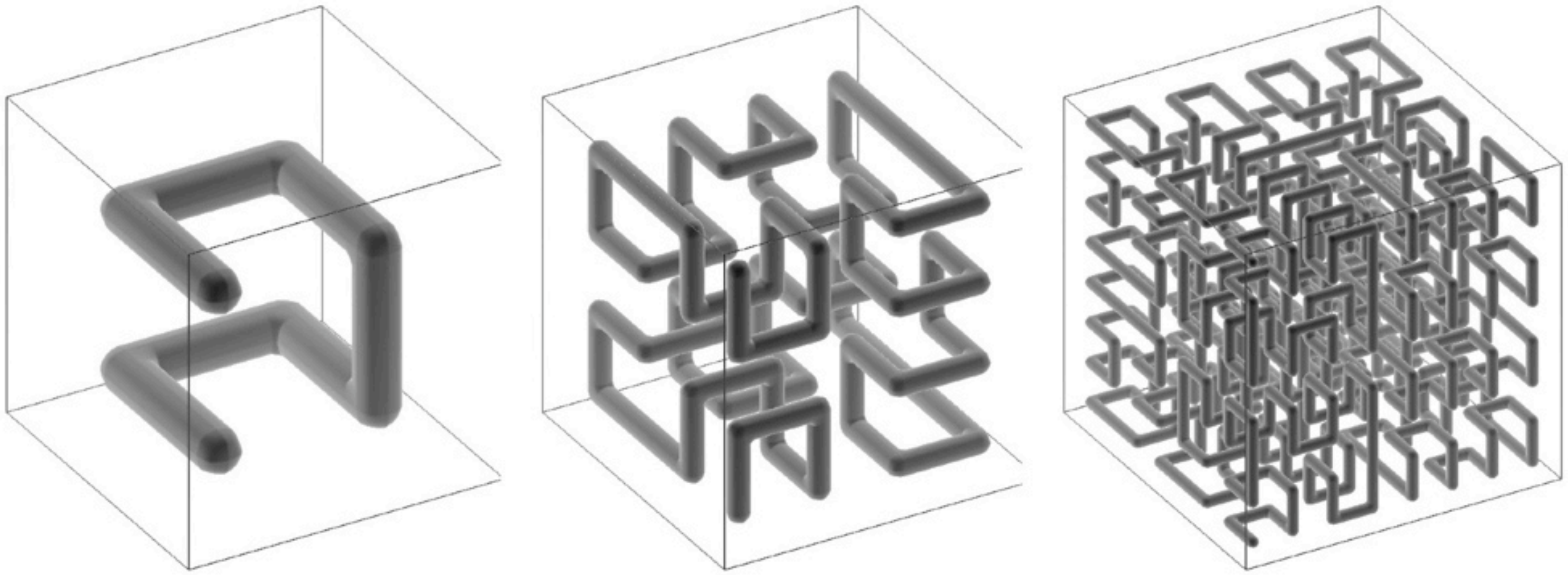
The remainder of Gadget compile-time and parameter file options .....

Compile-time options from within the Makefile:

```
OPT    +=   -DPEANOHILBERT
```

The space-filling Hilbert curve can be readily generalized to 3D

THE PEANO-HILBERT CURVE



3.9 The remaining parameters

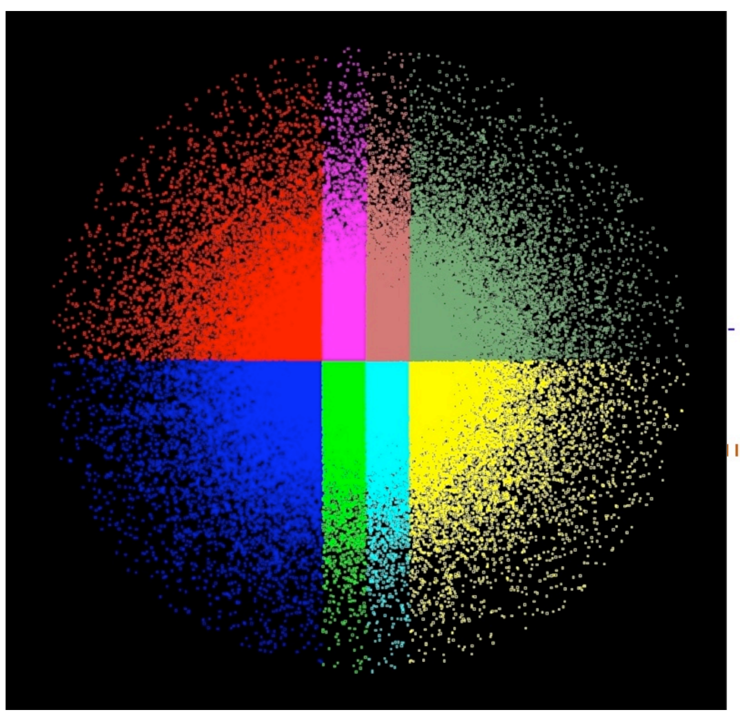
The remainder of Gadget compile-time and parameter file options .....

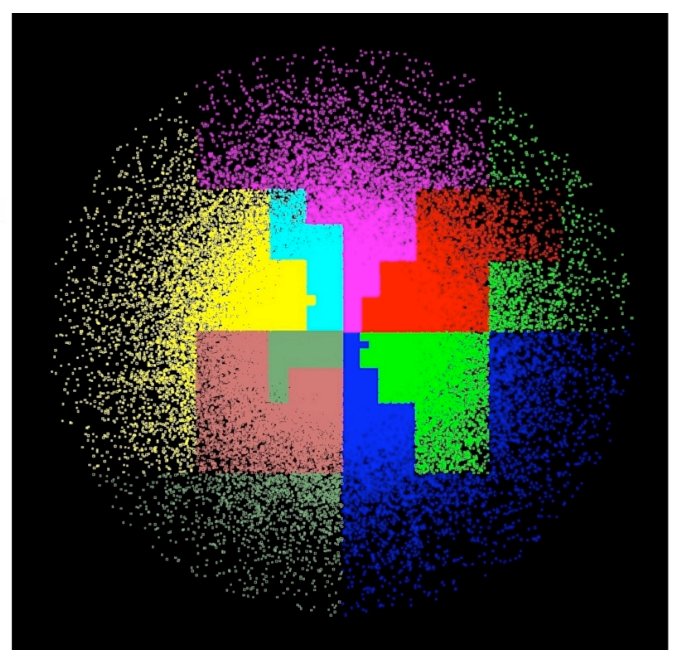
Compile-time options from within the Makefile:

```
OPT    +=   -DPEANOHILBERT
```

GADGET-1 used a simple orthogonal recursive bisection

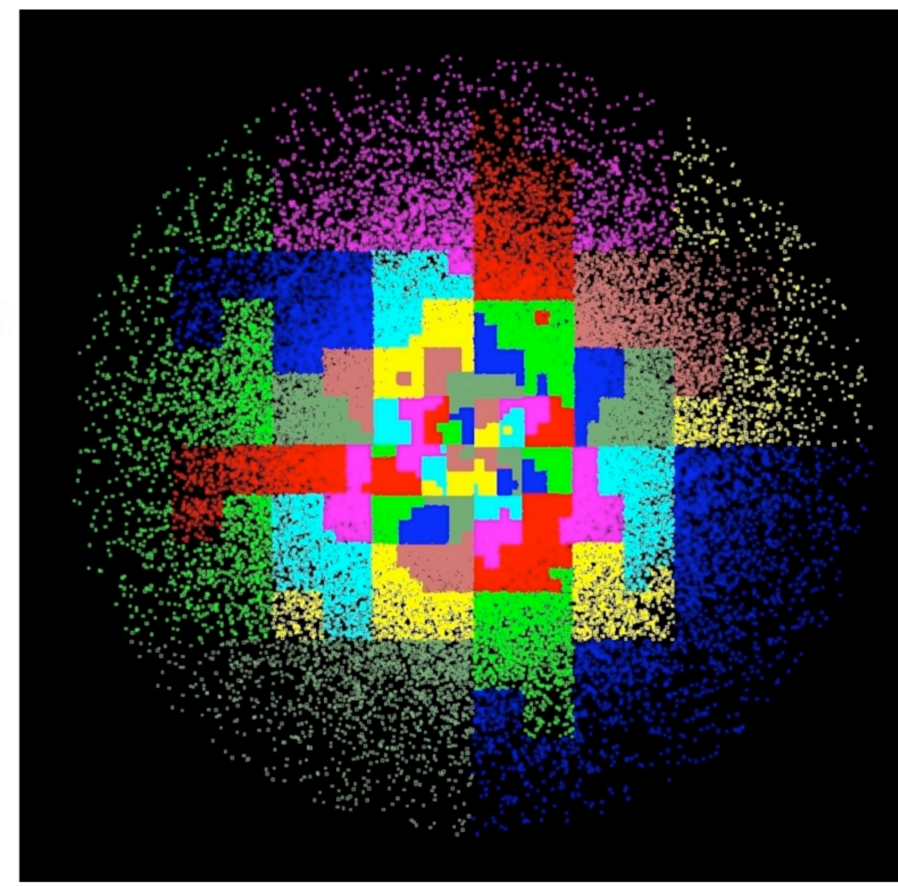
EXAMPLE OF DOMAIN DECOMPOSITION IN GADGET-1





GADGET-3 uses a space-filling Peano-Hilbert curve which is more flexible

EXAMPLE OF DOMAIN DECOMPOSITION IN GADGET-3



3.9 The remaining parameters

The remainder of Gadget compile-time and parameter file options .....

Parameter file options:

```
%  Relevant files

InitCondFile       /home/hipacc-5/MakeDiskGalaxy/Sa.dat
OutputDir          /home/hipacc-5/Sa/

EnergyFile         energy.txt
InfoFile           info.txt
TimingsFile        timings.txt
CpuFile            cpu.txt

RestartFile        restart
SnapshotFileBase   snapshot

OutputListFilename parameterfiles/output_list.txt
```

```
% Output frequency

TimeBetSnapshot        0.05
TimeOfFirstSnapshot    0

CpuTimeBetRestartFile     1800.0    ; here in seconds
TimeBetStatistics         0.05

NumFilesPerSnapshot       1
NumFilesWrittenInParallel 1
```

3.9 The remaining parameters

The remainder of Gadget compile-time and parameter file options .....

Parameter file options:

```
%  Relevant files

InitCondFile       /home/hipacc-5/MakeDiskGalaxy/Sa.dat
OutputDir          /home/hipacc-5/Sa/

EnergyFile         energy.txt
InfoFile           info.txt
TimingsFile        timings.txt
CpuFile            cpu.txt

RestartFile        restart
SnapshotFileBase   snapshot

OutputListFilename parameterfiles/output_list.txt
```

checkpoint

```
% Output frequency

TimeBetSnapshot        0.05
TimeOfFirstSnapshot    0

CpuTimeBetRestartFile     1800.0    ; here in seconds
TimeBetStatistics         0.05

NumFilesPerSnapshot       1
NumFilesWrittenInParallel 1
```

Restarting from a Checkpoint

PBS batchscript :

```
#!/bin/sh
#PBS -q batch
#PBS -N Sbc
#PBS -l nodes=2:ppn=2
#PBS -o Sbc.out
#PBS -e Sbc.err
#PBS -V
#PBS -M tcox@obs.carnegiescience.edu
#PBS -m abe

cd /home/hipacc-5/Sbc/

mpirun -v -machinefile $PBS_NODEFILE -np 4 ./Gadget2 Sbc.txt > output0.txt
```

add a "1" to restart from the last checkpoint - or you can add "2" to start from the last snapshot, but changes are also needed in the parameter file too

3.9 The remaining parameters

The remainder of Gadget compile-time and parameter file options .....

Parameter file options:

```
% CPU time -limit

TimeLimitCPU      36000  % = 10 hours
ResubmitOn        0
ResubmitCommand   my-scriptfile
```

3.9 The remaining parameters

The remainder of Gadget compile-time and parameter file options .....

Parameter file options:

```
ICFormat                1    }  with    #OPT   +=  -DHAVE_HDF5
SnapFormat              1
```

```
%  Caracteristics of run

TimeBegin           0.0       % Begin of the simulation
TimeMax             1.0       % End of the simulation
```

3.9 The remaining parameters

The remainder of Gadget compile-time and parameter file options .....

Parameter file options:

```
% Memory allocation

PartAllocFactor       3.5
TreeAllocFactor       1.5
BufferSize            50          % in MByte


% System of units

UnitLength_in_cm         3.085678e21        ;  1.0 kpc
UnitMass_in_g            1.989e43           ;  1.0e10 solar masses
UnitVelocity_in_cm_per_s 1e5                ;  1 km/sec
GravityConstantInternal  0
```

3.10 Odds and Ends

Questions?

How are any tests going?