Planetary Motion

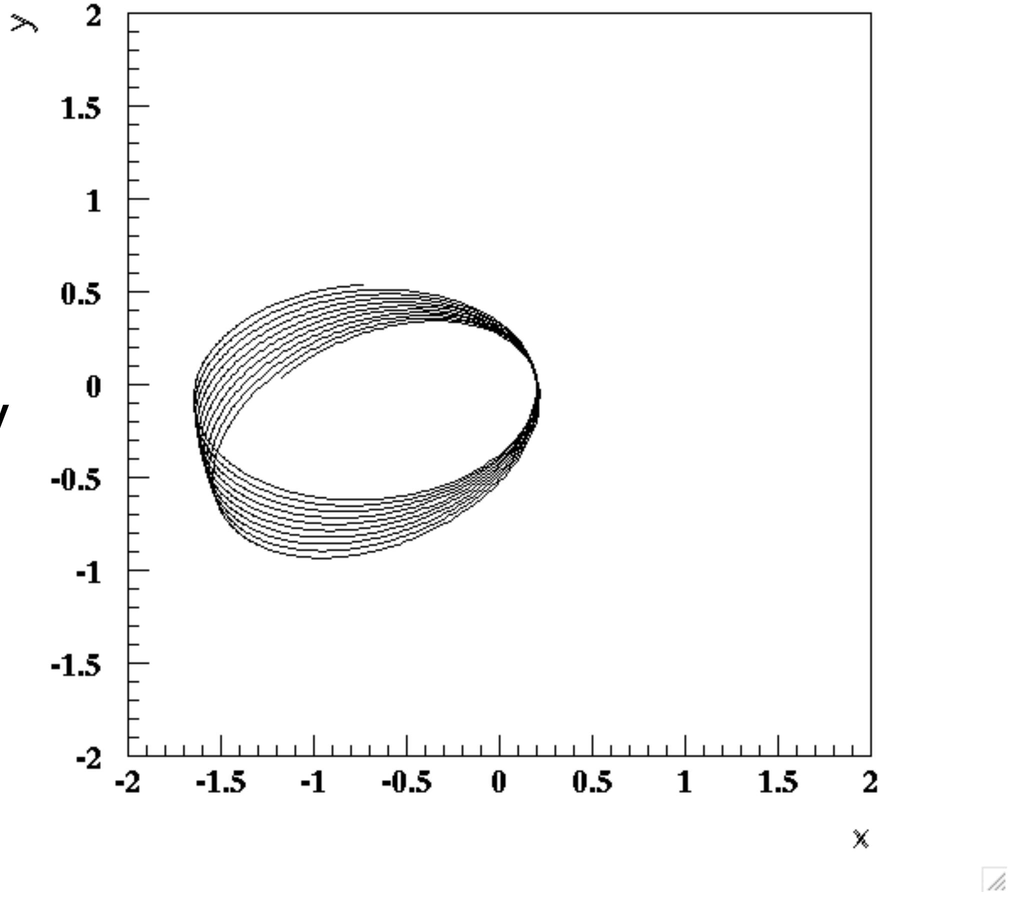
Last time, we discussed planetary motion for a 2-body system. We found that the closed orbits are given by ellipses. Now we take a closer look to see what happens if we change the force law.

This has a long history - Kepler’s equation led to Newton’s universal law of gravitation. Deviations from the 1/r^2 behavior indicate the presence of other bodies, and sometimes something more fundamental (General Relativity).

Planetary Motion

Runge-Kutta with adaptive step size for eps=0.0001 and for law $\propto r^{-2.01}$.

Axis of ellipse rotates with time (precession of the perihelion). Study of precession for Mercury has been historically very important.



Planetary Motion

So, it seems planetary motion could be used to test the force law. However, solar system has several planets + moons, so have to also look at effect of these.

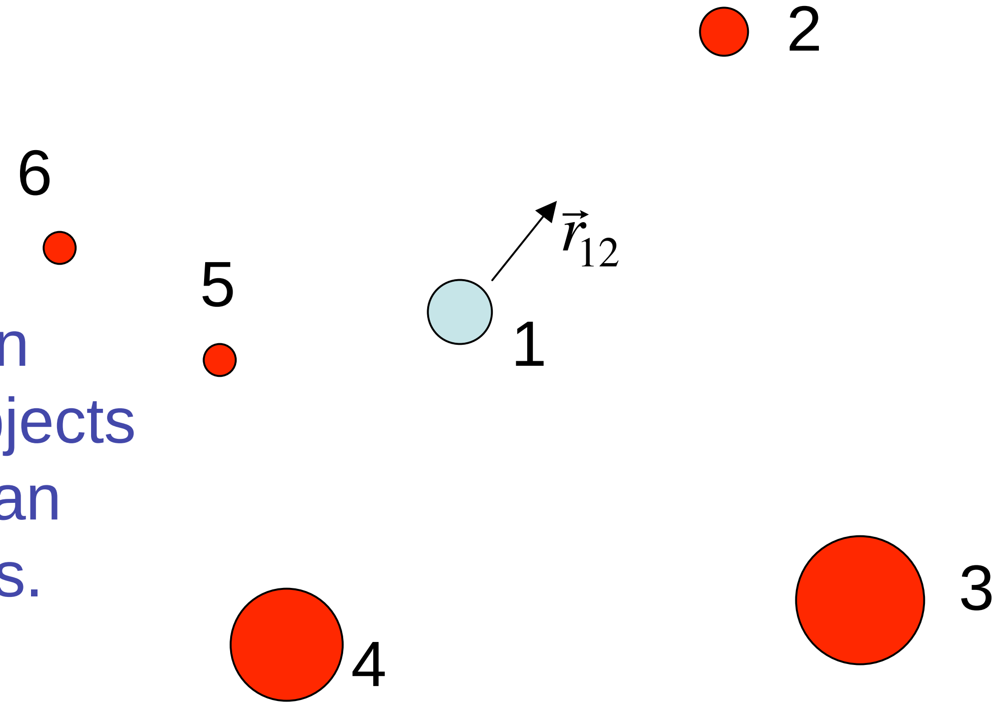
Mercury's perihelion makes one complete rotation every 230,000 years. Cannot be explained solely from the effect of other planets (will look at effect of Jupiter). Explanation came in 1917 from Einstein - General Relativity. Force law predicted by GR is of the form:

$$F_G \approx \frac{GMm}{r^2}\left(1+\frac{\alpha}{r^2}\right) \quad \text{with} \quad \alpha \approx 1.1 \cdot 10^{-8}\ \text{AU}^2$$

Planetary Motion

The n-body problem:

$$\vec{F}_i = \sum_{j \neq i}^{n} \frac{G m_i m_j}{r^3} \vec{r}_{ij} \quad \text{where } \vec{r}_{ij} \text{ is the vector from } i \text{ to } j$$



Need to define position and velocities of all objects at some time. Then can follow their interactions.

Planetary Motion

Try the 2nd order R-K method with adaptive time step for 3 objects interacting gravitationally.

Program outline:

1. Initialize masses, positions and velocities
2. Update positions and velocities using the 2nd order R-K method with step size D
3. Update positions and velocities using two steps each of size D/2.
4. Compare positions and velocities from 2.,3. If within range, accept, else reduce time step and go back to 2.
5. Stop after fixed time

For this, we write a general program to update the positions and velocities using the 2nd order R-K method.

Planetary Motion

$$x(i,j;t+\Delta t)=x(i,j;t)+f_{x(i,j)}(\{\vec{x}'\},\{\vec{v}'\},t')\Delta t$$

$$v(i,j;t+\Delta t)=v(i,j;t)+f_{v(i,j)}(\{\vec{x}'\},\{\vec{v}'\},t')\Delta t$$

i is the particle index
j is the coordinate x,y,z

$$f_{x(i,j)}(\{\vec{x}\},\{\vec{v}\},t)=v(i,j)$$

$$f_{v(i,j)}(\{\vec{x}\},\{\vec{v}\},t)=\sum_{k\neq i}\frac{Gm_k(x(k,j)-x(i,j))}{\sum_{j=1}^{3}\left|x(i,j)-x(k,j)\right|^3}$$

Update of position depends only on velocity component.

Update of velocity depends to total force in that direction.

and

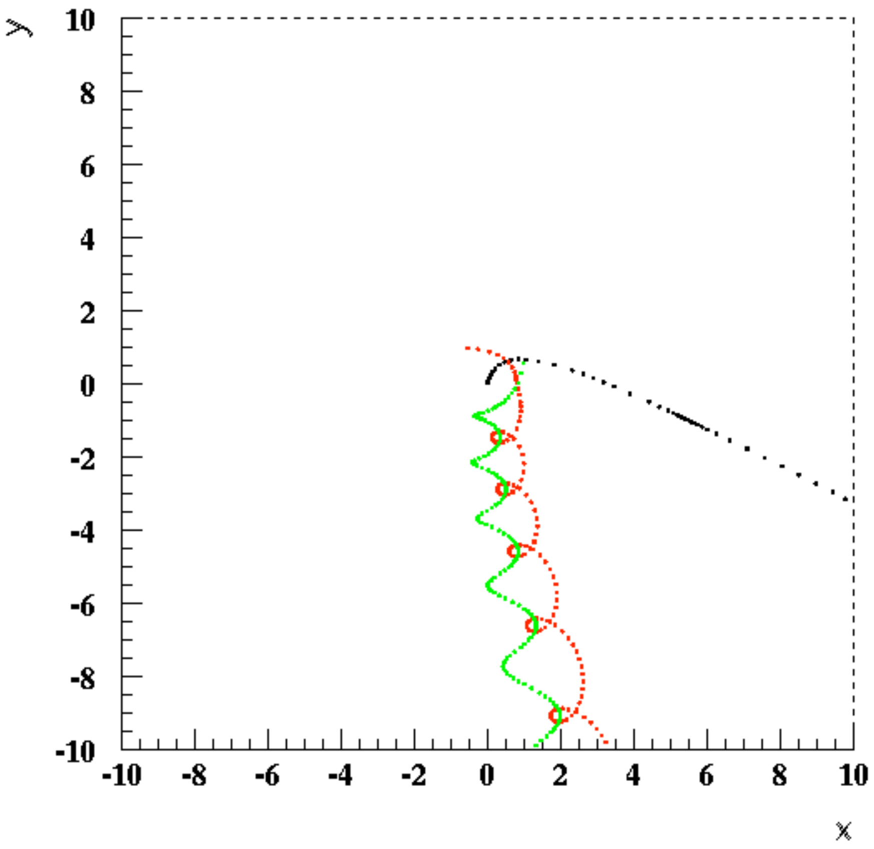
$$x'(i,j)=x(i,j)+\frac{1}{2}v(i,j)\Delta t$$

$$v'(i,j)=v(i,j)+\frac{1}{2}\sum_{k\neq i}\frac{Gm_k(x(k,j)-x(i,j))}{\sum_{j=1}^{3}\left|x(i,j)-x(k,j)\right|^3}\Delta t$$

$$t'=t+1/2\Delta t$$

Planetary Motion

An example:

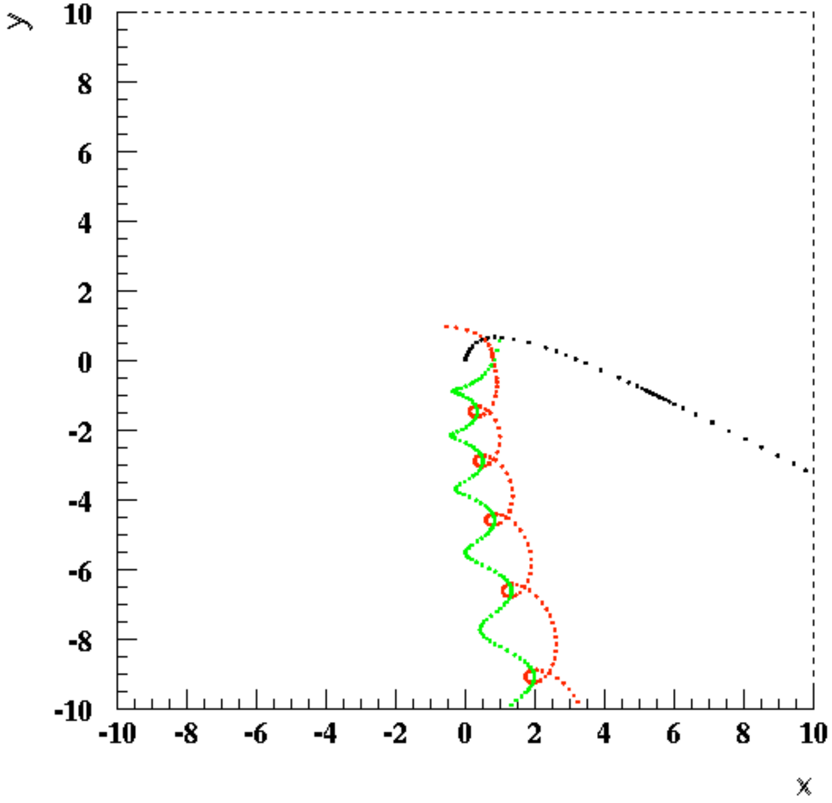


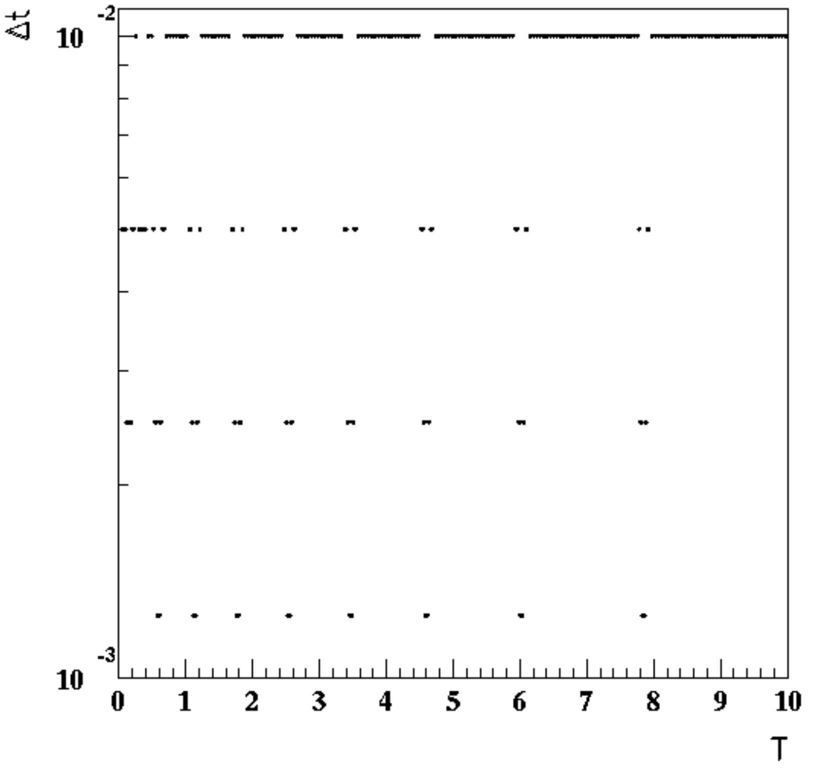
Initial conditions:
dt=0.01 (years)

$Gm_1=Gm_2=Gm_3=4\pi^2$

x(1,1)=0.	v(1,1)=0.
x(2,1)=-1.	v(2,1)=10.
x(3,1)=1.	v(3,1)=0.
x(1,2)=0.	v(1,2)=0.
x(2,2)=1.	v(2,2)=0.
x(3,2)=1.	v(3,2)=-10.
x(1,3)=0.	v(1,3)=0.
x(2,3)=0.	v(2,3)=0.
x(3,3)=0.	v(3,3)=0.

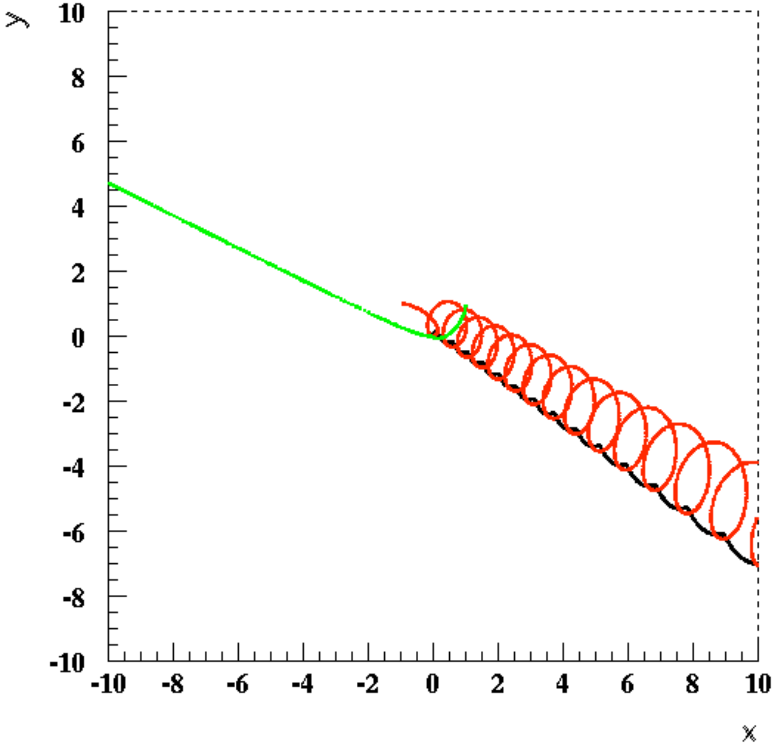
Planetary Motion





Planetary Motion

Different example

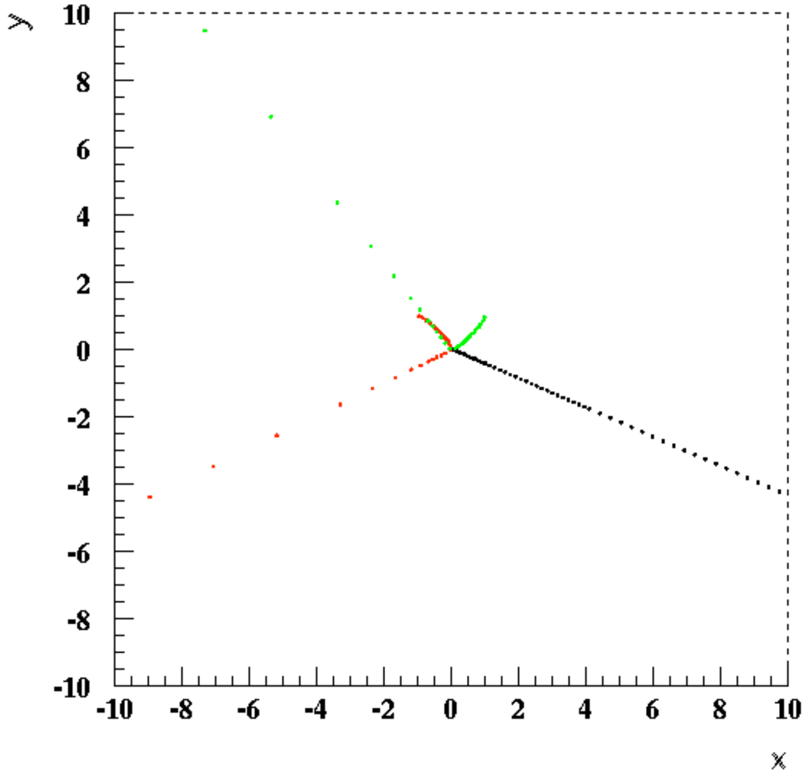


Initial conditions:
dt=0.01 (years)

$Gm_1=40\pi^2$
$Gm_2=Gm_3=4\pi^2$

x(1,1)=0.	v(1,1)=0.
x(2,1)=-1.	v(2,1)=10.
x(3,1)=1.	v(3,1)=0.
x(1,2)=0.	v(1,2)=0.
x(2,2)=1.	v(2,2)=0.
x(3,2)=1.	v(3,2)=-10.
x(1,3)=0.	v(1,3)=0.
x(2,3)=0.	v(2,3)=0.
x(3,3)=0.	v(3,3)=0.

Planetary Motion



Initial conditions:
dt=0.01 (years)

$Gm_1=400\pi^2$
$Gm_2=Gm_3=4\pi^2$

x(1,1)=0.	v(1,1)=0.
x(2,1)=-1.	v(2,1)=10.
x(3,1)=1.	v(3,1)=0.
x(1,2)=0.	v(1,2)=0.
x(2,2)=1.	v(2,2)=0.
x(3,2)=1.	v(3,2)=-10.
x(1,3)=0.	v(1,3)=0.
x(2,3)=0.	v(2,3)=0.
x(3,3)=0.	v(3,3)=0.

Numerical breakdown: objects get very close together - very large forces, huge velocities.

Other Approaches to the N-body Problem

There are many approaches to the N-body problem See e.g., http://www.amara.com/papers/nbody.html.

- Particle-Particle (PP)
- Particle-Mesh (PM)
- Particle-Particle/Particle-Mesh (P3M)
- Particle Multiple-Mesh (PM2)
- Nested Grid Particle-Mesh (NGPM)
- Tree-Code (TC) Top Down
- Tree-Code (TC) Bottom Up
- Fast-Multipole-Method (FMM)
- Tree-Code Particle Mesh (TPM)
- Self-Consistent Field (SCF)
- Symplectic Method

Particle-Particle Approach

The Particle-Particle method is the method we just looked at:

1. Accumulate forces by finding the force F(i,j) of particle j on particle i,
2. Integrate the equations of motion (which includes the accumulated forces), and
3. Update the time counter.
4. Repeat for the next time step.

In our approach, we performed the simulation in double precision with an adaptive time step. Still ran into numerical problems. We only used 2nd order R-K. I have seen references where 7th order R-K is used.

Particle-Mesh Approach

The Particle-Mesh method treats the force as a field quantity by approximating it on a mesh. Differential operators, such as the laplacian, are replaced by finite difference approximations. Potentials and forces at particle positions are obtained by interpolating on the array of mesh-defined values. Mesh-defined densities are calculated by assigning particle attributes (e.g. "charge") to nearby mesh points in order to create the mesh-defined values (e.g. "charge density"). This is closely related to the next problem we will address.

So the principle steps of the particle mesh calculation are:

1. Assign "charge" to the mesh ("particle mass" becomes "grid density"),
2. Solve the field potiential equation (e.g. Poisson's) on the mesh,
3. Calculate the force field from the mesh-defined potential,
4. Interpolate the force on the grid to find forces on the particles.
5. Now like the PP: integrate the forces to get particle positions and velocities.
6. Update the time counter.

Partial Differential Equations

A very common numerical application is solving partial differential equations with boundary conditions. E.g., Laplace's equation:

$$\frac{\partial^2 V}{\partial x^2}+\frac{\partial^2 V}{\partial y^2}+\frac{\partial^2 V}{\partial z^2}=0$$

This is somewhat different that the previous problems, in that we are trying to find a static solution, rather than look at the time evolution of a system. Also, the boundary conditions are typically the value of *V* on a surface, rather than specifying single points.

There are no standard techniques, such as the Euler method or Runge-Kutta to solve the problem numerically. However, a general method, called 'relaxation' often works.

Relaxation Methods

Relaxation methods typically work well for elliptic equations …

General approach:

1. Define a grid where we will solve for the potential at the grid points.
2. Define the boundary conditions on the grid. Values have to be fixed also for the unknown interior points.
3. Update the grid values using the differential equations. Keep repeating the update until the grid values are stable within a prescribed value.

The main issues are:

- choosing an appropriate grid
- specifying the initial values on the grid
- using a good updating algorithm

Relaxation Methods

We have seen in Lecture 5 that we can approximate a second derivative as follows:

$$\frac{d^2 y}{dx^2} \approx \frac{y(x+\Delta x) - 2y(x) + y(x-\Delta x)}{(\Delta x)^2}$$

With this approximation, Laplace’s approach becomes:

$$\frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2} + \frac{\partial^2 V}{\partial z^2} \approx \frac{V(i+1,j,k) - 2V(i,j,k) + V(i-1,j,k)}{(\Delta x)^2} +$$

$$\frac{V(i,j+1,k) - 2V(i,j,k) + V(i,j+1,k)}{(\Delta y)^2} + \frac{V(i,j,k+1) - 2V(i,j,k) + V(i,j,k+1)}{(\Delta z)^2} = 0$$

With solution ….

$$V(i,j,k) = \frac{\left[(V(i+1,j,k) + V(i-1,j,k)\right](\Delta y)^2(\Delta z)^2 + \ldots}{2\left((\Delta y)^2(\Delta z)^2 + (\Delta x)^2(\Delta z)^2 + (\Delta x)^2(\Delta y)^2\right)}$$

Relaxation Methods

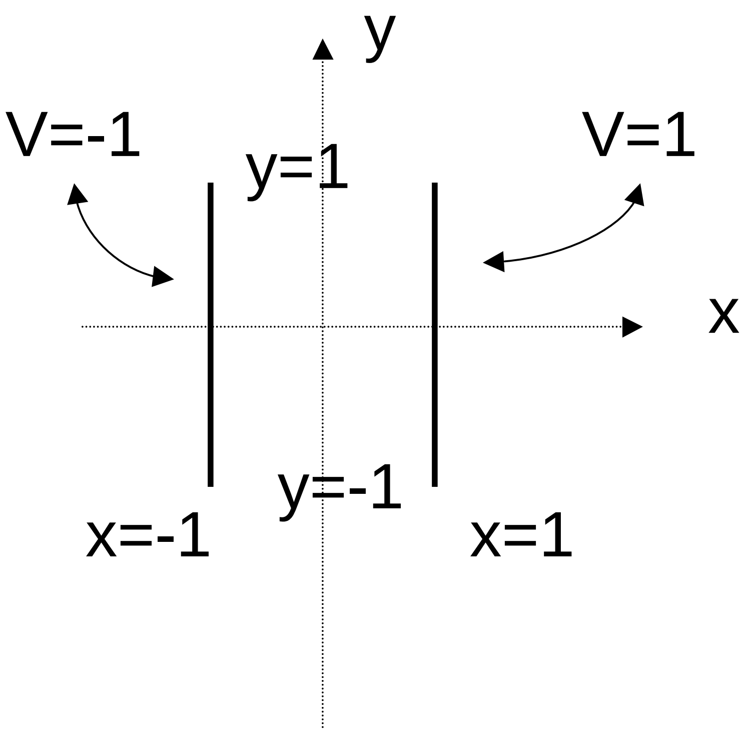
For equal step sizes in the three dimension, we have:

$$V(i,j,k) = \frac{1}{6}\begin{bmatrix} V(i+1,j,k) + V(i-1,j,k) + V(i,j+1,k) + \\ V(i,j-1,k) + V(i,j,k+1) + V(i,j,k-1) \end{bmatrix}$$

So, we update the value of the potential at a point using the values at the neighboring sites. Note that the boundary values are not allowed to be updated ! It is the boundary values that specify the final solution - they are propagated by the differential equations through the rest of the volume.

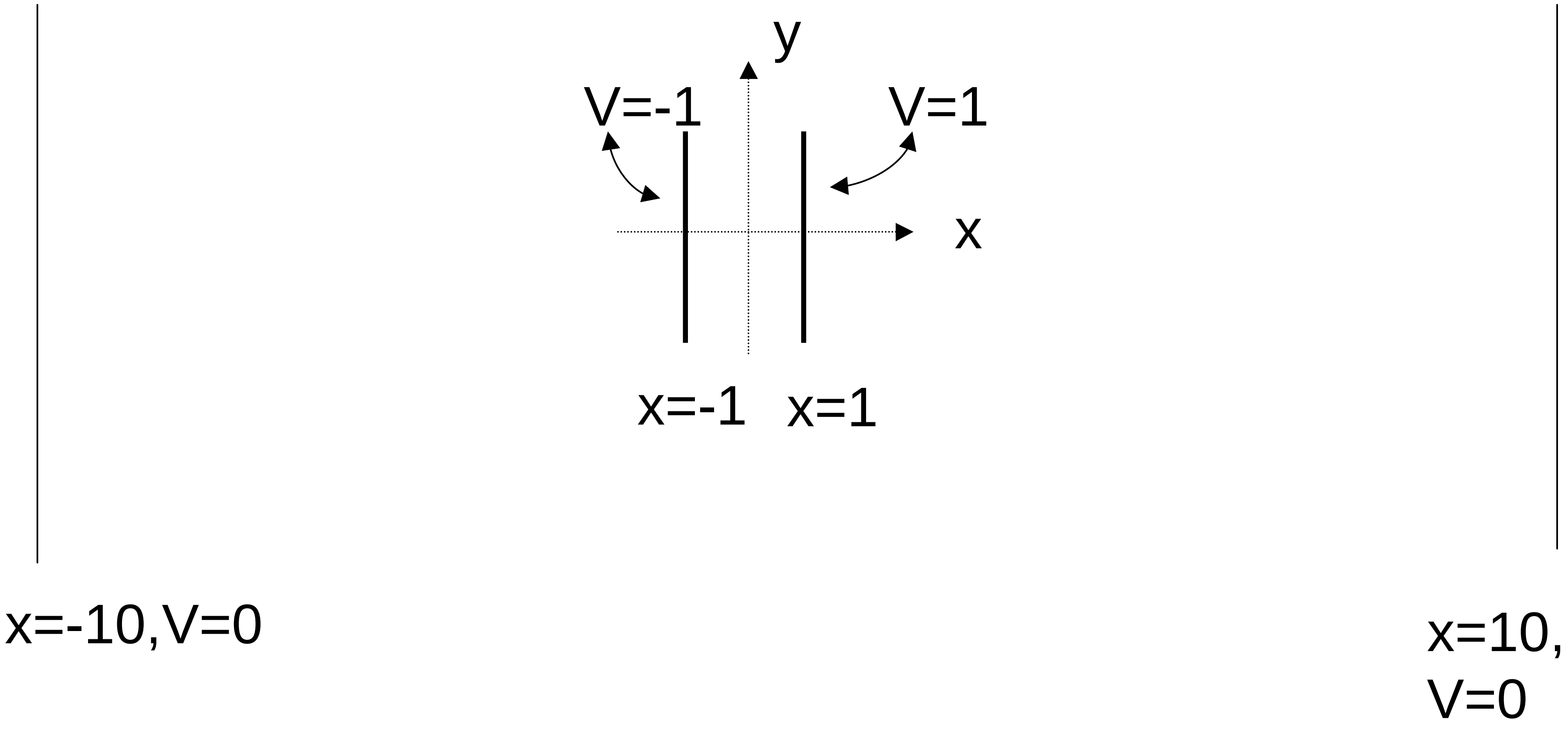
In problems with symmetry, choose coordinate system in such a way that the calculational problem can be reduced. Try a parallel plate capacitor as an example.

Parallel Plate Capacitor



If the plates are very long in the z direction, then near the center of the capacitor we can treat this as a 2-dimensional problem. Apart from the value of the potential on the capacitor surfaces, we also need the potential far away. Set it to zero on a square at |x|=10.,|y|=10. Define the grid in such a way that the boundaries are on grid points.

Parallel Plate Capacitor



Number of grid points along x: $(x_{max}-x_{min})/\Delta x + 1$ is an integer
Further requirement $(x_{max}-1)/ \Delta x$ is an integer. Similarly for y
Let's take $\Delta x=0.2$, $\Delta y=0.2$, for a total of $(101)^2=10201$ grid points.

Parallel Plate Capacitor

```
* Initialize the grid - start with zero everywhere
      Call Vzero(V,10201)        Need two arrays for the potential V(101,101),V0(101,101)
      Call Vzero(V0,10201)
      Call Vzero(Ex,10201)       And two arrays for the electric field components
      Call Vzero(Ey,10201)
* Now fill in the boundary conditions.
      x=-1.                      Start with the left plate
      Nx=(x-xmin)/xstep+1        Find the index given our step size
      Do Ny=1,Nygrid             Loop over y grid positions
        y=ymin+(Ny-1)*ystep      Calculate the y value at this grid position
        If (abs(y).le.1.) then   See if it is in the range of our capacitor plate
          V(Nx,Ny)=-1.           Yes, then define the potential here
        Endif
      Enddo
      x=+1.                      Now the right plate, same procedure
      Nx=(x-xmin)/xstep+1
      Do Ny=1,Nygrid
        y=ymin+(Ny-1)*ystep
        If (abs(y).le.1.) then
          V(Nx,Ny)=+1.
        Endif
      Enddo
```

Parallel Plate Capacitor

```
      Do Nx=1,Nxgrid
        Do Ny=1,Nygrid
          V0(Nx,Ny)=V(Nx,Ny)          Copy the grid to our second array
        Enddo
      Enddo
* Loop over the grid points and update the potential.
2     Iteration = Iteration+1         Keep track of the iteration number
      diffmax=0.                      This variable will keep track of the biggest change in V
*
      Do Nx=2,Nxgrid-1                Note that we leave off the end points - cant change BC
        x=xmin+(Nx-1)*xstep
*
        Do Ny=2,Nygrid-1,1
          y=ymin+(Ny-1)*ystep
*
          If (x.eq.-1. .and. (abs(y).le.1.) ) goto 1   Check if on one of the plates
          If (x.eq.1. .and. (abs(y).le.1.) ) goto 1
* Here we update the potential (2 dimensions because of symmetry)
          V(Nx,Ny)=(V(Nx+1,Ny)+V(Nx-1,Ny)+V(Nx,Ny+1)+V(Nx,Ny-1))/4.
*
```

Parallel Plate Capacitor

```
* Compare to previous value - keep track of maximum change
           If (abs(V(Nx,Ny)-V0(Nx,Ny)).gt.diffmax)   diffmax=abs(V(Nx,Ny)-V0(Nx,Ny))
* Update the electric field

          Ex(Nx,Ny)=(V(Nx+1,Ny)-V(Nx-1,Ny))/(2.*xstep)   Centered difference
          Ey(Nx,Ny)=(V(Nx,Ny+1)-V(Nx,Ny-1))/(2.*ystep)    Recall that E is a vector
*
 1           continue
          Enddo
       Enddo                     End of loop over grid points
* Copy updated value of array

      Do Nx=1,Nxgrid
          Do Ny=1,Nygrid
             V0(Nx,Ny)=V(Nx,Ny)
          Enddo
       Enddo
* Check if we have converged or timed out
       If (diffmax.gt.eps .and. Iteration.lt.Itermax) goto 2
```

Parallel Plate Capacitor

Parameters:

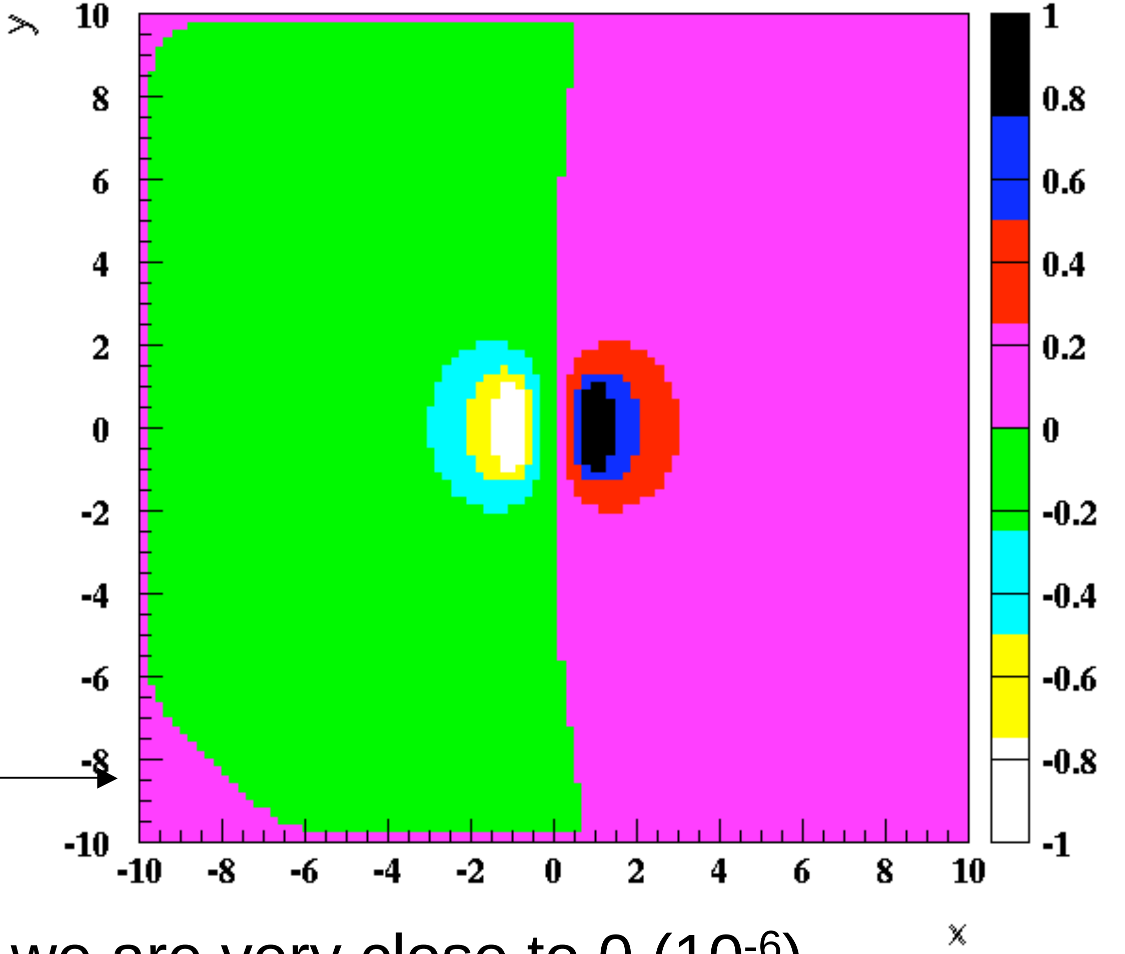
```
       Data xstep/0.2/,ystep/0.2/,xmin/-10./,xmax/10./,
     &     ymin/-10./,ymax/10./
       Data eps/0.001/
       Data Vwall/0./,Vcapacitor1/-1./,Vcapacitor2/1./
       Data Nxgrid/101/,Nygrid/101/
       Data Itermax/1000/
*
```

The program required 139 iterations to converge !

Next time (+1), we will look into more efficient algorithms. Let’s look at the results:

Parallel Plate Capacitor

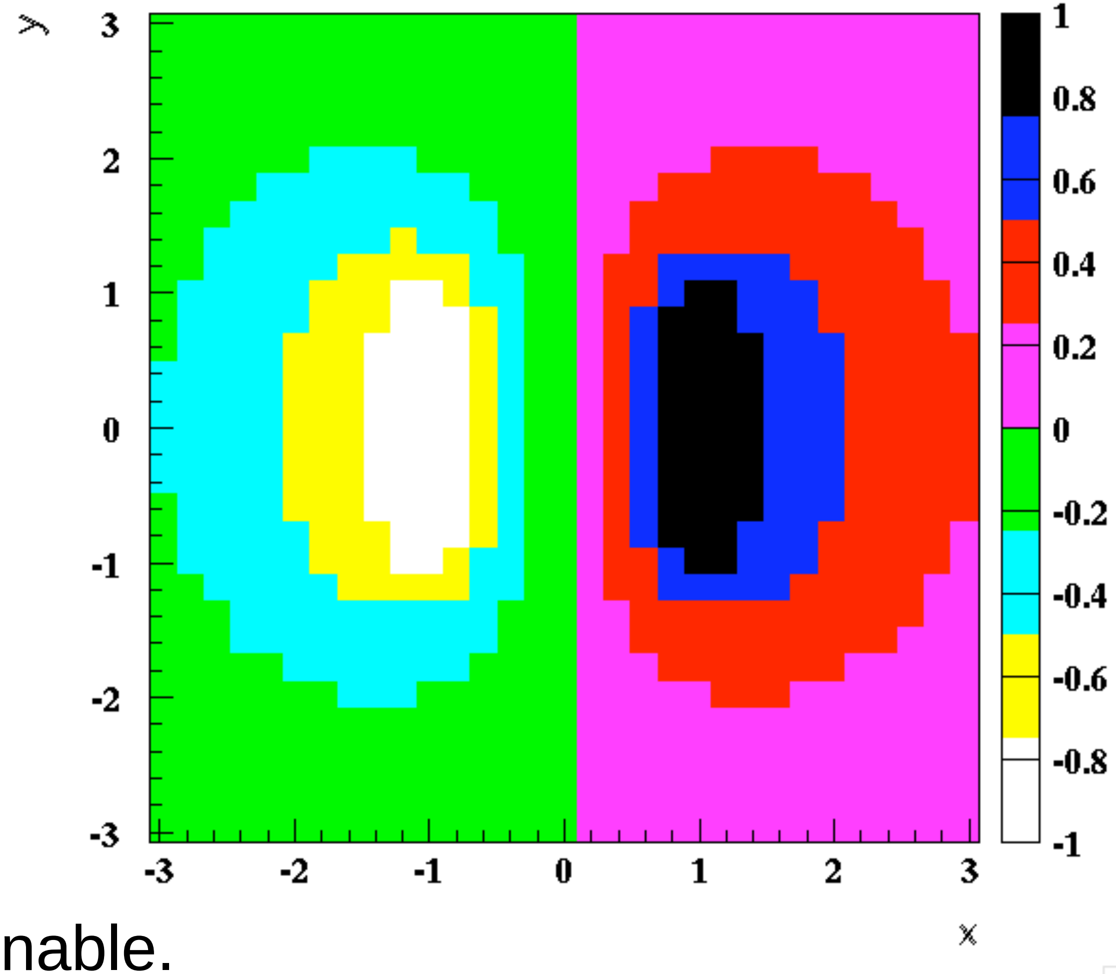
Potential



Here we are very close to 0 (10^{-6}).

Parallel Plate Capacitor

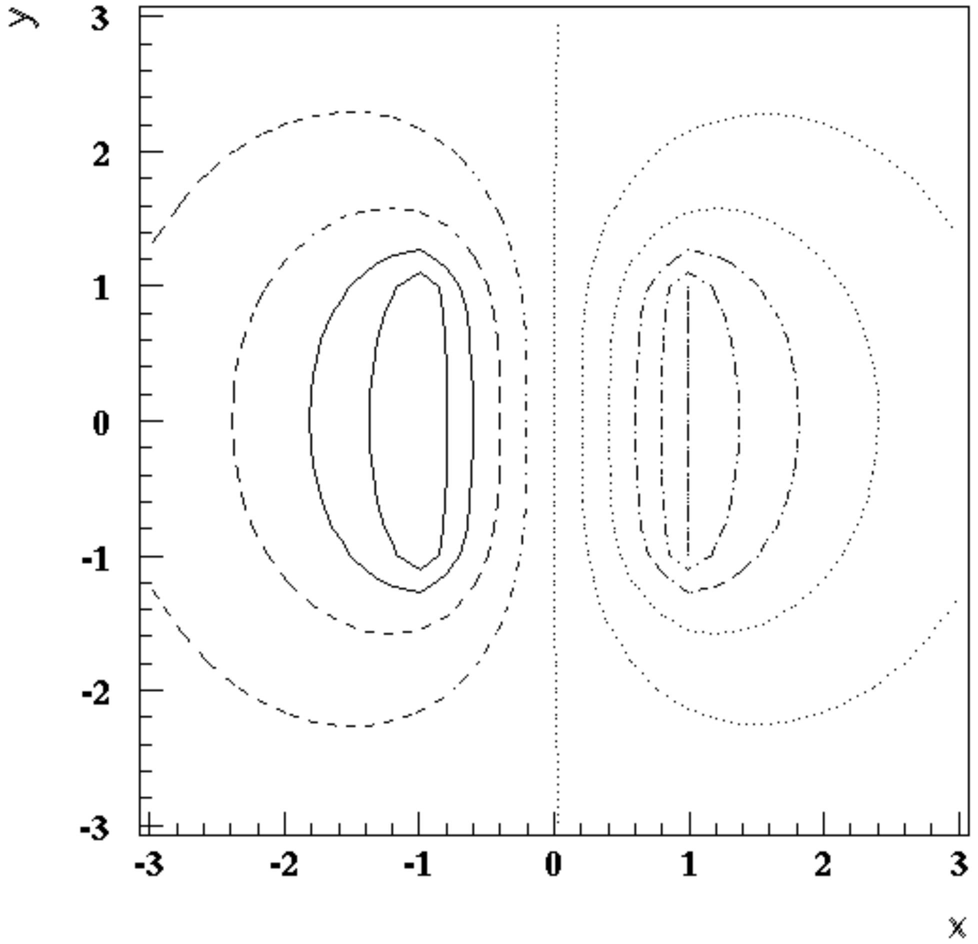
Zoom in on region near the plates:



Looks reasonable.

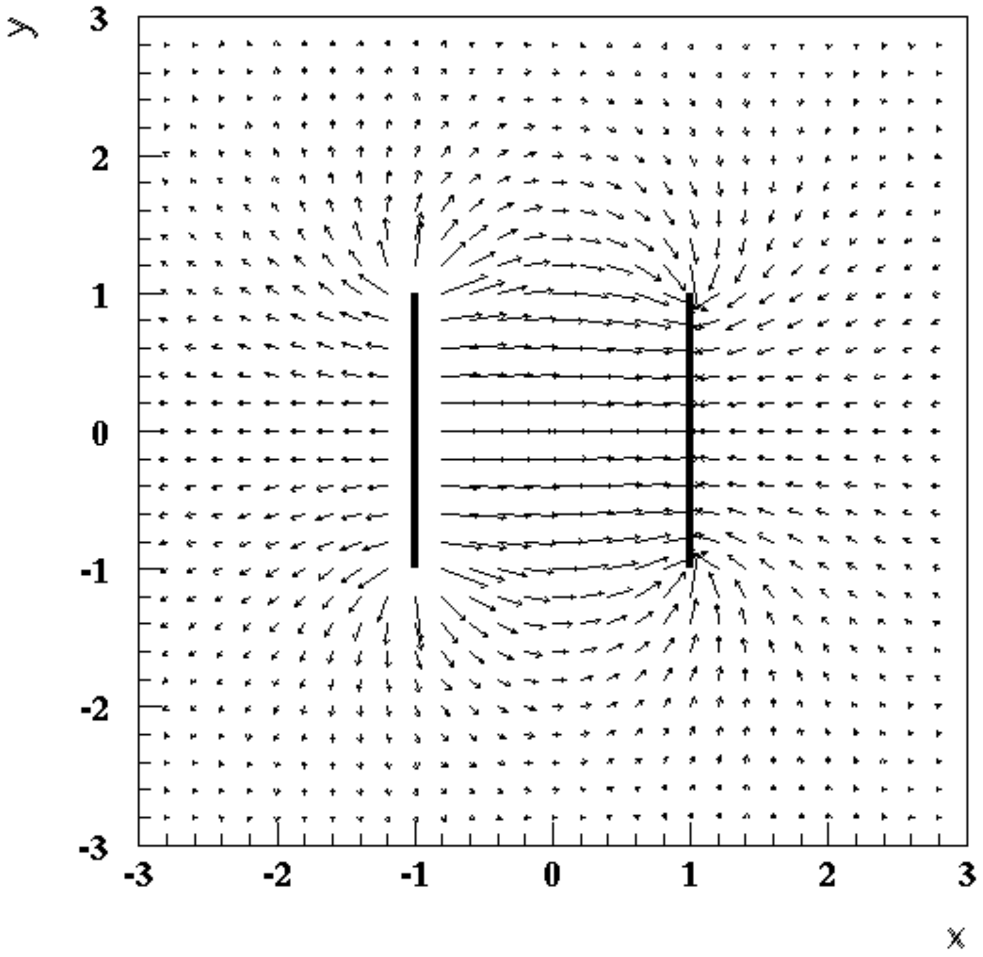
Parallel Plate Capacitor

A different presentation:



Parallel Plate Capacitor

Now for the electric field. Length of arrow gives strength of field. Direction from E_x and E_y



Exercizes

1. Write a program to simulate the motion of three particles interacting gravitationally using the Euler method with adaptive time step. Try it out with different initial conditions and plot the results.

2. Find the potential and electric field numerically for the space between two concentric cylinders, the inner cylinder at V=-1 and the outer cylinder at V=+1. Assume the cylinders are very long and consider the region near the center of the cylinders.