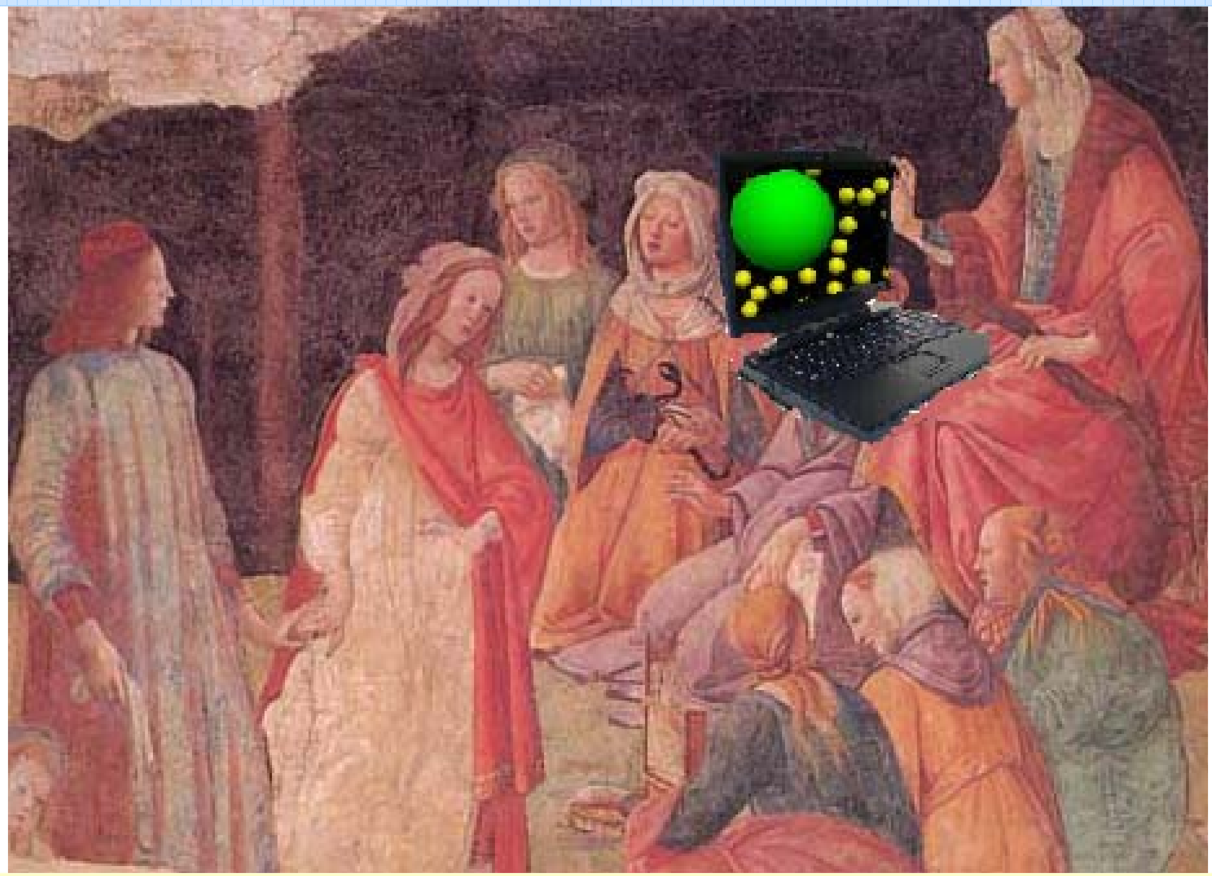
AAPT Topical Conference:
Computational Physics for Upper Level Courses
Davidson College, 2007

Integrating Computation and Research into the Liberal Arts Physics Curriculum



Amy Bug
Dept. of Physics and Astronomy, Swarthmore College

Note to the reader …

Most, but not all, of these slides were shown on 7-27-07 at the AAPT satellite meeting at Davidson College. You may take and use any of the slides to forward the good cause of integrating computation into the college physics curriculum. I welcome any feedback you have about them.

Additionally, if you would like some of the curricular materials that I've used with Swarthmore students, you can contact me and I will direct you to a website, burn you a CD and mail it to you, or both. Available are:
⇒Assignments and more from the 1999 version of the comp. phys. course
⇒ Assignments and more from the 2006 version of the comp. phys. course
⇒MATLAB and Mathematica* notes, assigned problems, and some solutions from our sophomore level math methods lab

Amy Bug, 500 College Ave. Swarthmore College, Swarthmore, PA 19081
abug1@swarthmore.edu

** Our Mathematica notes and assigned problems come straight from Nick Wheeler at Reed College. Perhaps he is distributing them, and has a more recent version than we do.*

A liberal arts college ...

trains its students to do nothing

and prepares its students to do everything. *-anon.*

A liberal arts college …

trains its students to do nothing

and prepares its students to do everything. -anon.

NB: This talk is about the upper-level physics curriculum , not a “liberal arts physics” course aimed at non-science folks.

A liberal arts college …

trains its students to do nothing

and prepares its students to do everything. -anon.

Swarthmore Physics/Astro **Class of 2007** (17 majors) …

Advanced degrees in
- Physics
- Astronomy
- C.S.
- Math
- Archaeology

Jobs
- Nanofabrication
- Financial sector
- K-12 teaching
- English teaching abroad
- C.S.
- Peace Corps

Undecided

A liberal arts college …

trains its students to do nothing

and prepares its students to do everything. -anon.

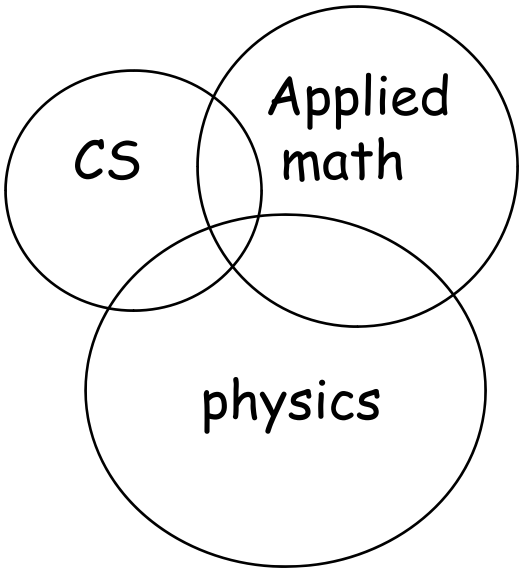
Swarthmore Physics/Astro **Class of 2007** (17 majors) …

- Advanced degrees in
  - Physics
  - Astronomy
  - C.S.
  - Math
  - Archaeology
- Jobs
  - Nanofabrication
  - Financial sector
  - K-12 teaching
  - English teaching abroad
  - C.S.
  - Peace Corps
- Undecided

…has a commitment to
- interdisciplinary work
- diversity

Why teach students computational physics?

Interdisciplinary:



Why teach students computational physics?

Historical:

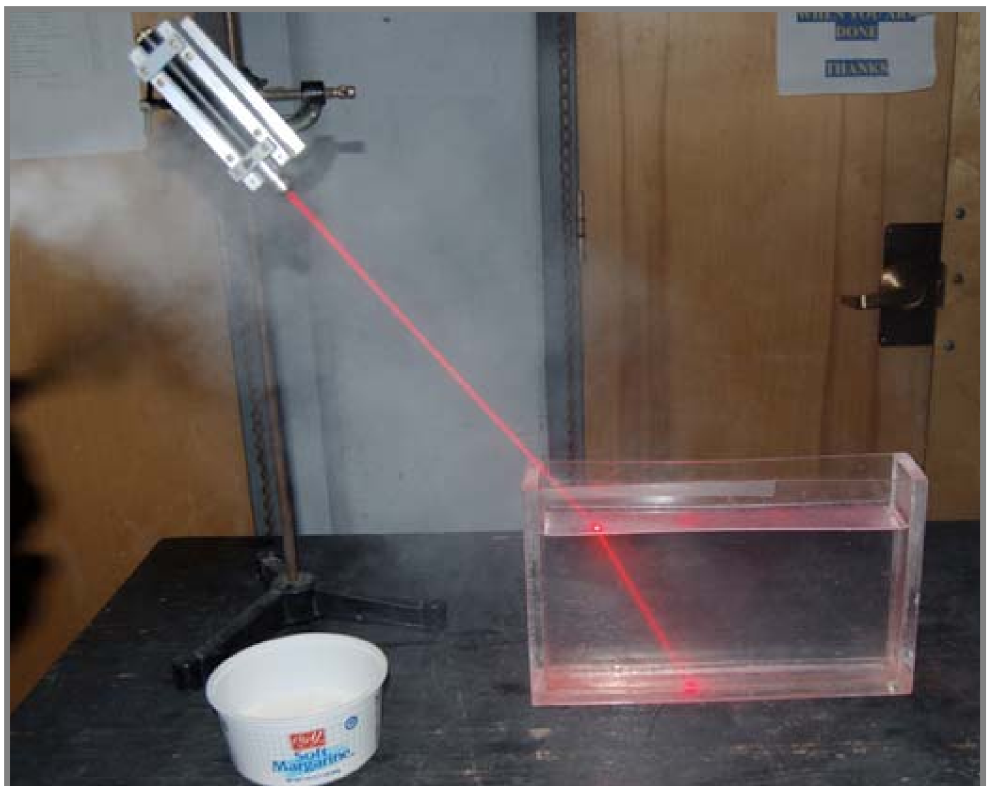
- *Lord Kelvin (1901)*
- *Fermi, Metropolis, … (1930's, 1940's)*
- *Feigenbaum (1980's)*

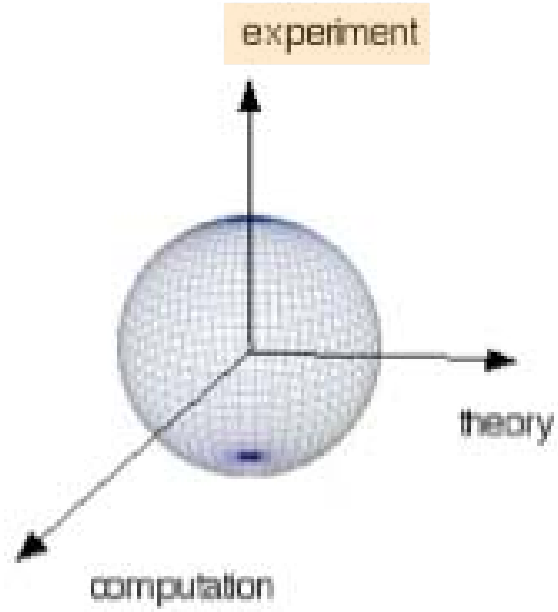
QuickTime™ and a
TIFF (Uncompressed) decompressor
are needed to see this picture.

The women of Eniac

Why teach students computational physics?

Builds intuition about the physical world

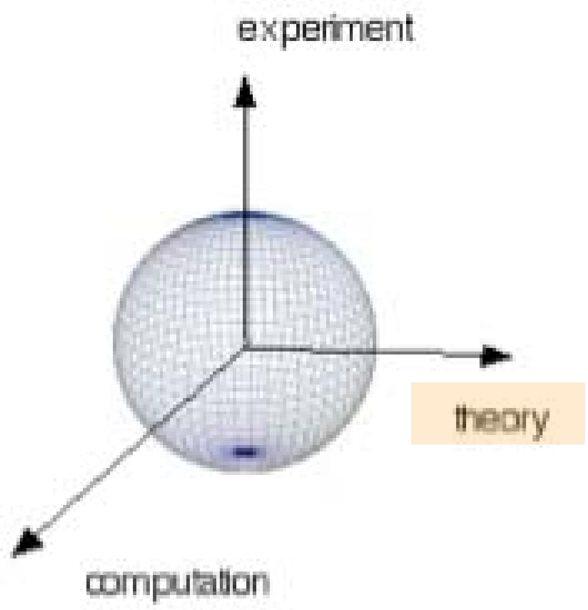




What happens?

Why teach students computational physics?

Builds intuition about the physical world



$$T = \int_{t_A}^{t_B} dt$$

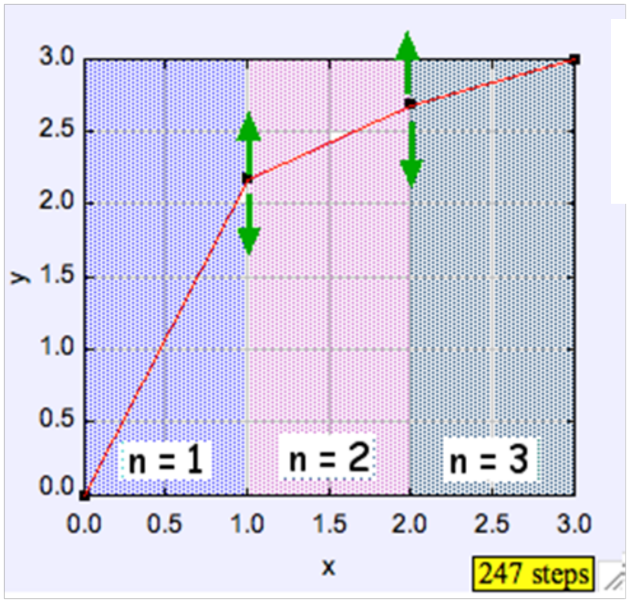
$$= \int_{x_A, y_A}^{x_B, y_B} (ds/dt)^{-1} ds$$

$$\delta T = 0 \; ; \quad x_A, y_A, x_B, y_B \quad \text{fixed}$$

Why?

Why teach students computational physics?

Builds intuition about the physical world

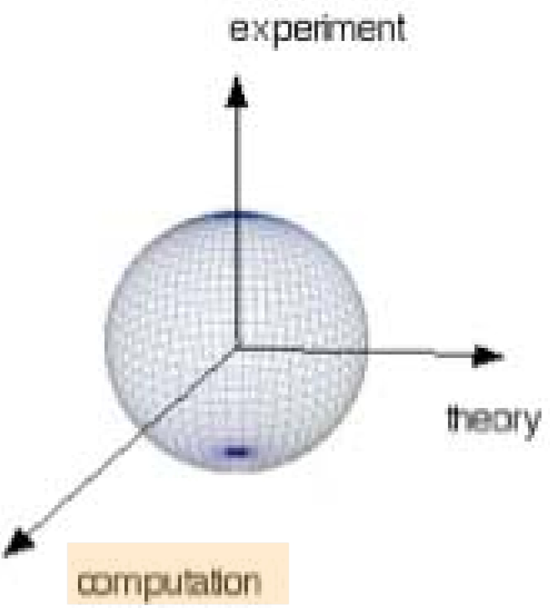


$$T = \int_{t_A}^{t_B} dt$$

$$\approx \sum_i d_i / v_i$$

$$= \sum_i d_i n_i / c$$

Nodes between media execute ***random walk*** *in y direction. If new path involves shorter T, old path is replaced by new. (G,T & C, 2006)*



What and why?

Why teach students computational physics?

Builds intuition about the physical world

Builds generic problem-solving skills

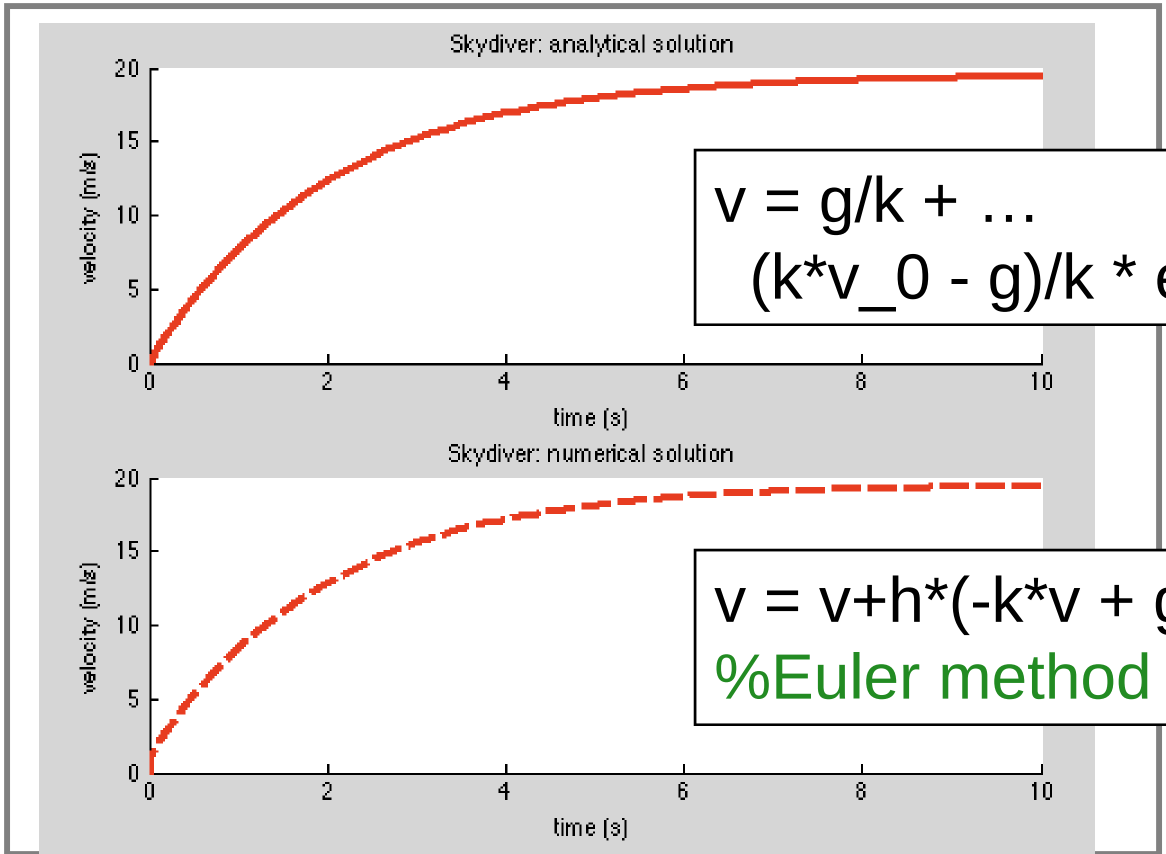
- *Use of appropriate units*
- *Model building*
- *Make approximations and test limiting behaviors*
- *Visualization*
- *Error analysis*
- *…*

Why teach students computational physics?

Builds intuition about the physical world

Builds generic problem-solving skills

Can explore/visualize families of solutions, analytical or numerical




```
v = g/k + ...
  (k*v_0 - g)/k * exp(-k*t)
```

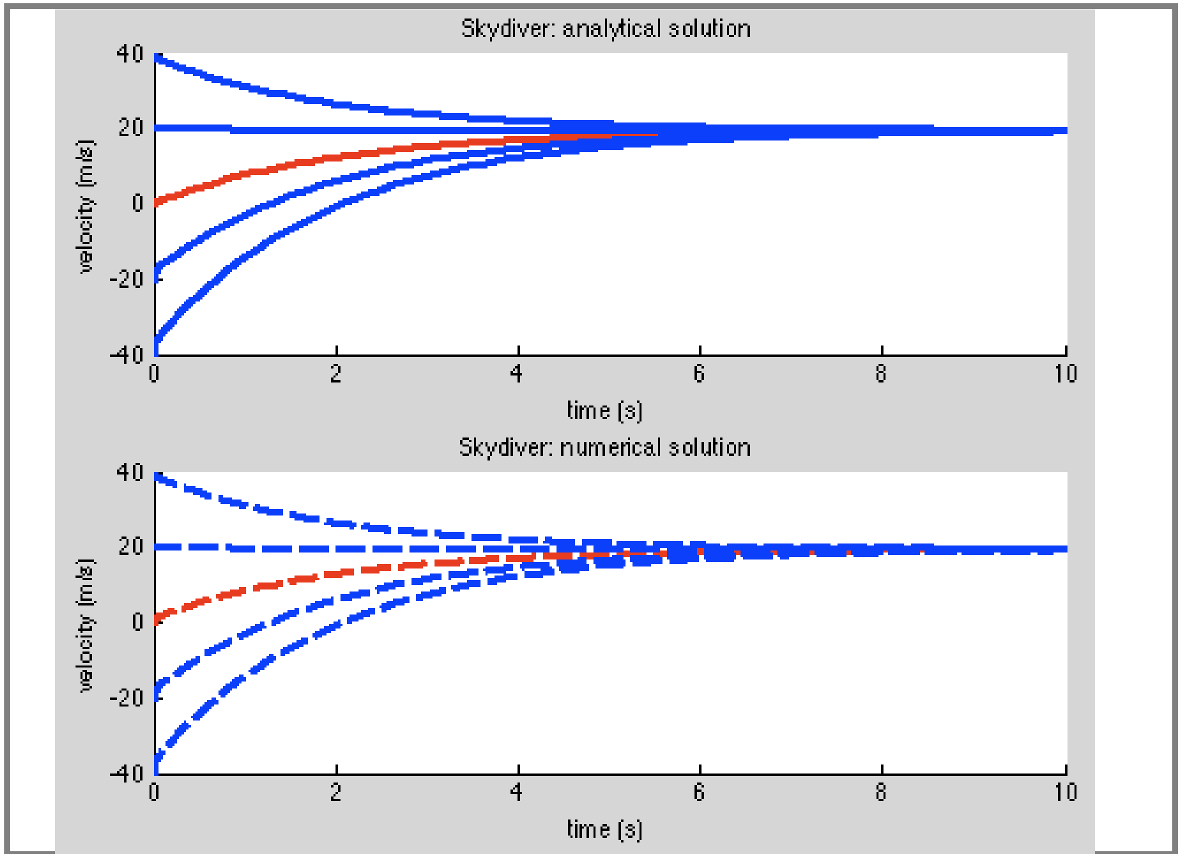
```
v = v+h*(-k*v + g);
%Euler method
```

Why teach students computational physics?

Builds intuition about the physical world

Builds generic problem-solving skills

Can explore/visualize families of solutions, analytical or numerical



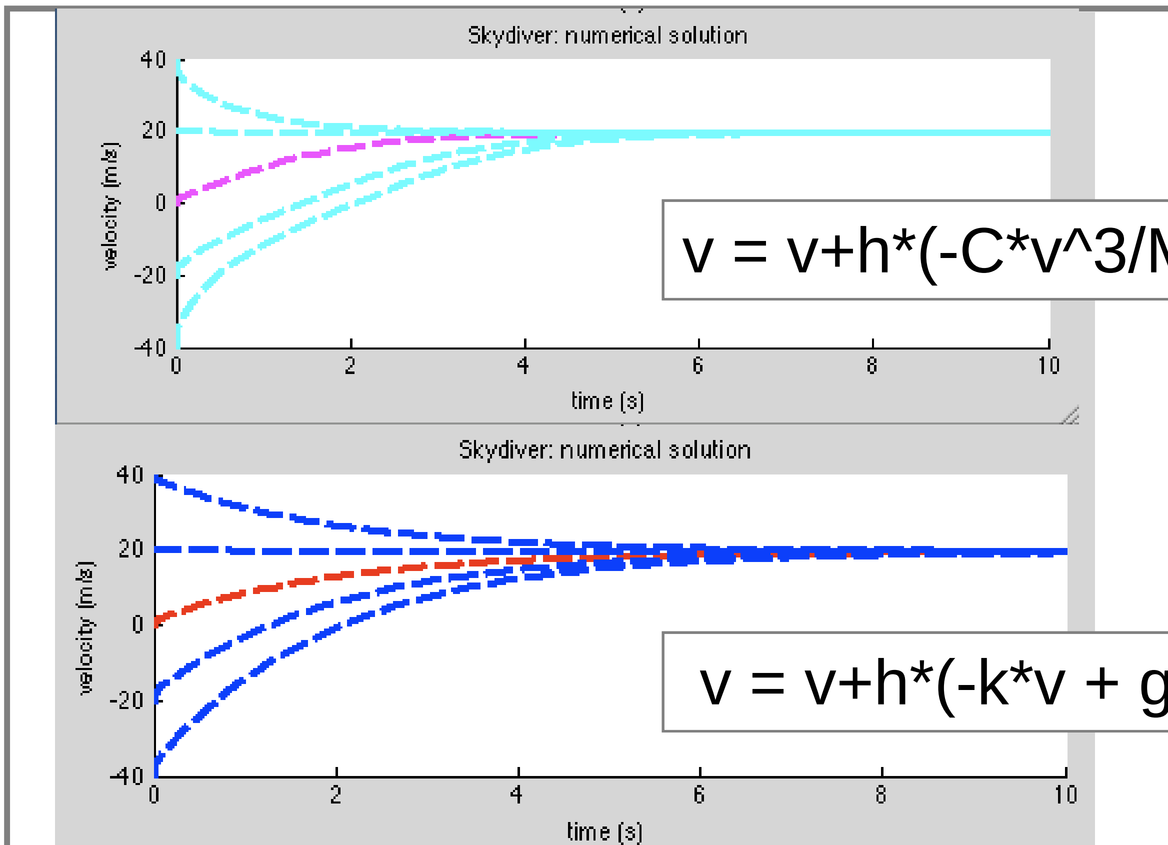
Why teach students computational physics?

Builds intuition about the physical world

Builds generic problem-solving skills

Can explore/visualize families of solutions, analytical or numerical

Can enlarge the class of accessible problems




```
v = v+h*(-C*v^3/M + g);
```

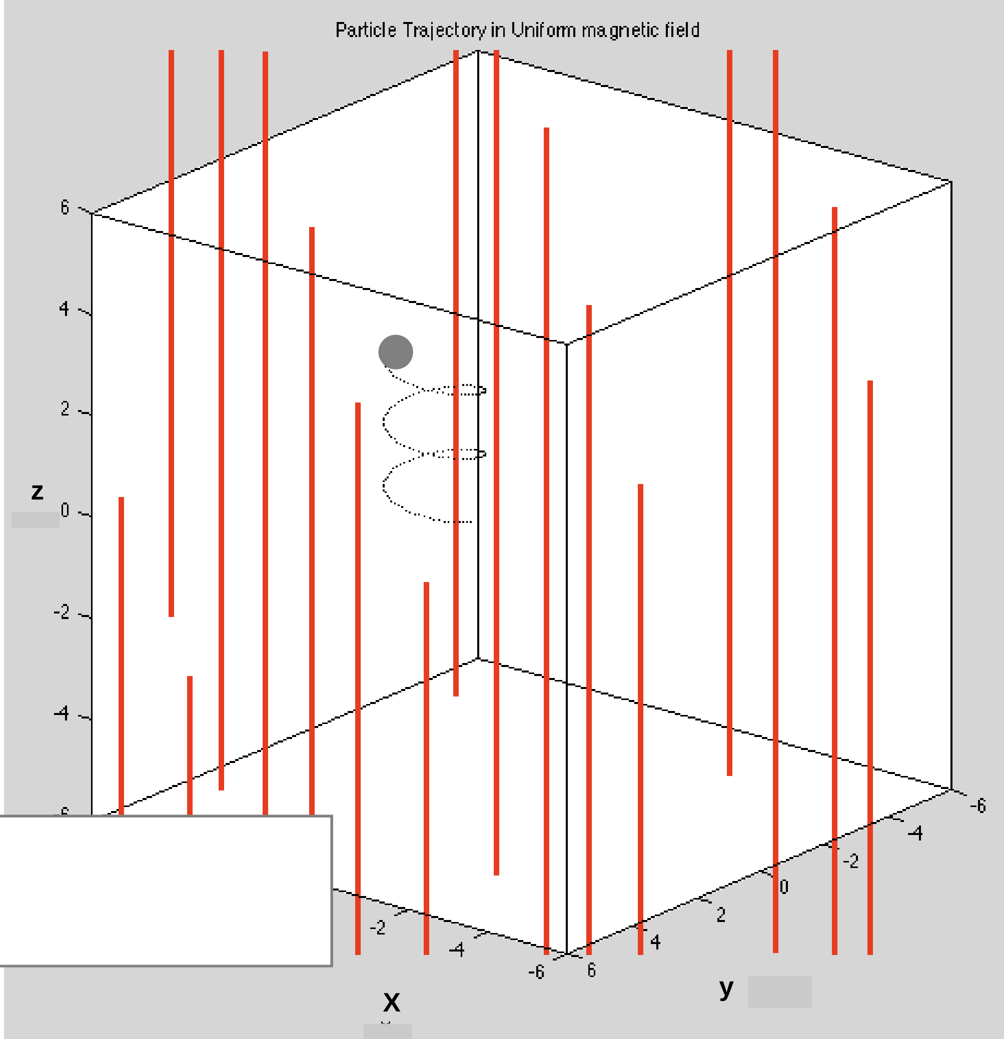
```
v = v+h*(-k*v + g);
```

Why teach students computational physics?

Builds intuition about the physical world

Builds generic problem-solving skills

Can explore/visualize families of solutions, analytical or numerical




```
function a=accelx(Bz,x,y,z,vx,vy,vz)
a=(vy*Bz);
```

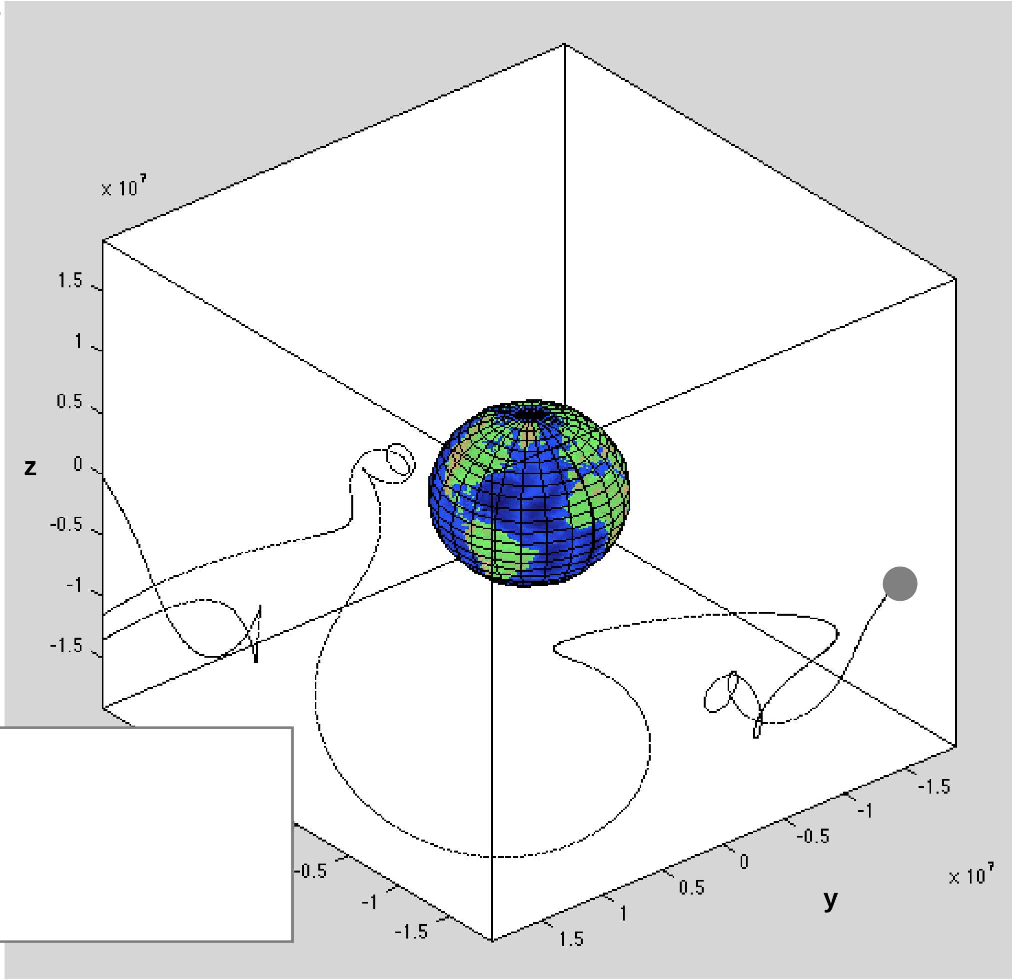
Why teach students computational physics?

Builds intuition about the physical world

Builds generic problem-solving skills

Can explore/visualize families of solutions, analytical or numerical

Can enlarge the world of accessible problems and find novel behaviors.




```
function a=accelx(r,x,y,z,vx,vy,vz)
a=((2*z*z-x*x-y*y)*vy- ...
        3*y*z*vz)/r^5;
```

At our college, computational physics manifests as …

Lower level

- ❖ *tool for the professor during* ***lecture***
- ❖ *occasional activity for students in* ***lab***

Upper level: required

- ❖ *part of a sophomore spring* ***math methods course***

Upper level: elective

- ❖ *tool for students in* ***upper level (seminar) courses***
- ❖ *an occasionally-offered junior/senior* ***seminar***
- ❖ *a tool in student/faculty* ***experimental research***
- ❖ *a student/faculty* ***research area***

At our college, computational physics manifests as …

Lower level

- *tool for the professor during* ***lecture***
- *occasional activity for students in* ***lab***

Upper level: required

- *part of a sophomore spring* ***math methods course***

Upper level: elective

- *tool for students in* ***upper level (seminar) courses***
- *an occasionally-offered junior/senior* ***seminar***
- *a tool in student/faculty* ***experimental research***
- *a student/faculty* ***research area***

At our college, computational physics manifests as …

Lower level

❖ *tool for the professor during* ***lecture***

❖ *occasional activity for students in* ***lab***

Upper level: required

❖ *part of a sophomore spring* ***math methods course***

Upper level: elective

❖ *tool for students in* ***upper level (seminar) courses***

❖ *an occasionally-offered junior/senior* ***seminar***

❖ *a tool in student/faculty* ***experimental research***

❖ *a student/faculty* ***research area***

*note that computing is **not** …*

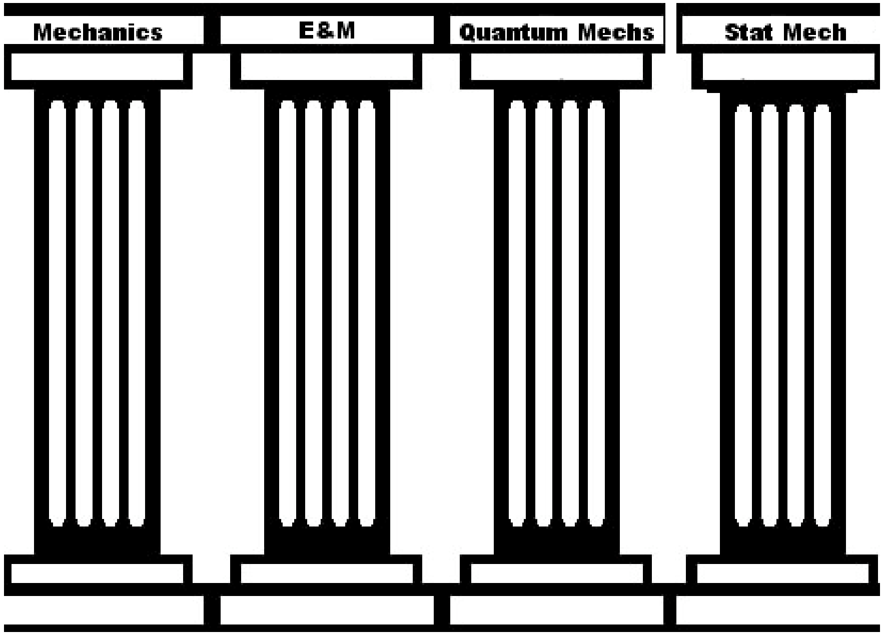
❖ *used by all faculty teaching a core subject*

❖ *required of students, save in the math methods course*

❖ *something all faculty members feel comfortable doing/teaching*

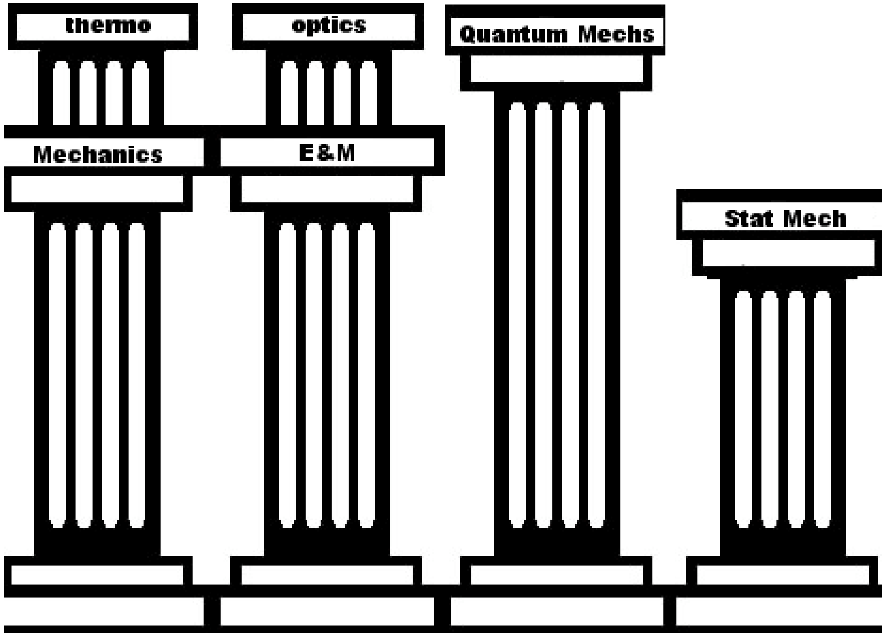
Curriculum: the "teach core subjects and teach them twice" model …

Four pillars:



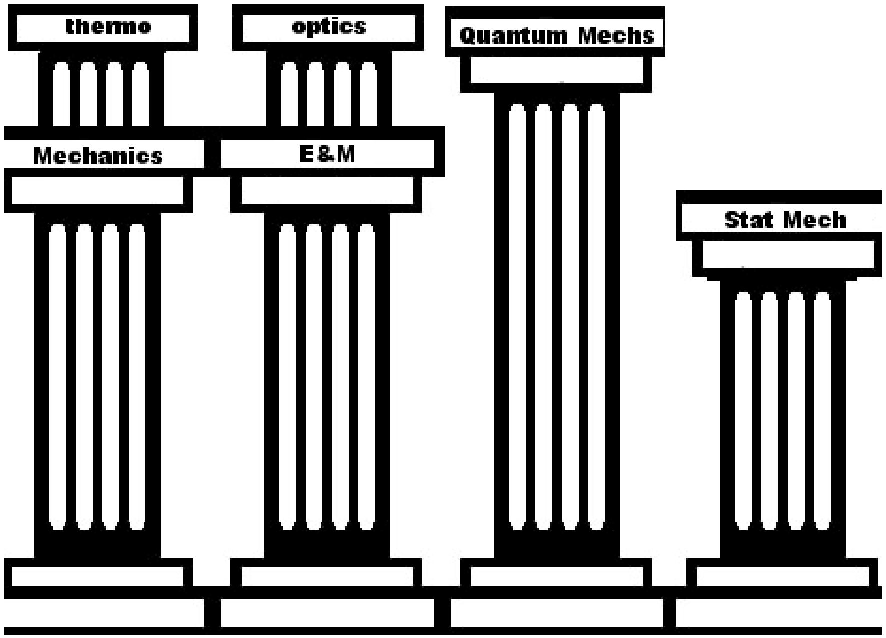
Curriculum: the "teach a few subjects and teach them twice" model …

Four pillars:



Curriculum: the "teach a few subjects and teach them twice" model …

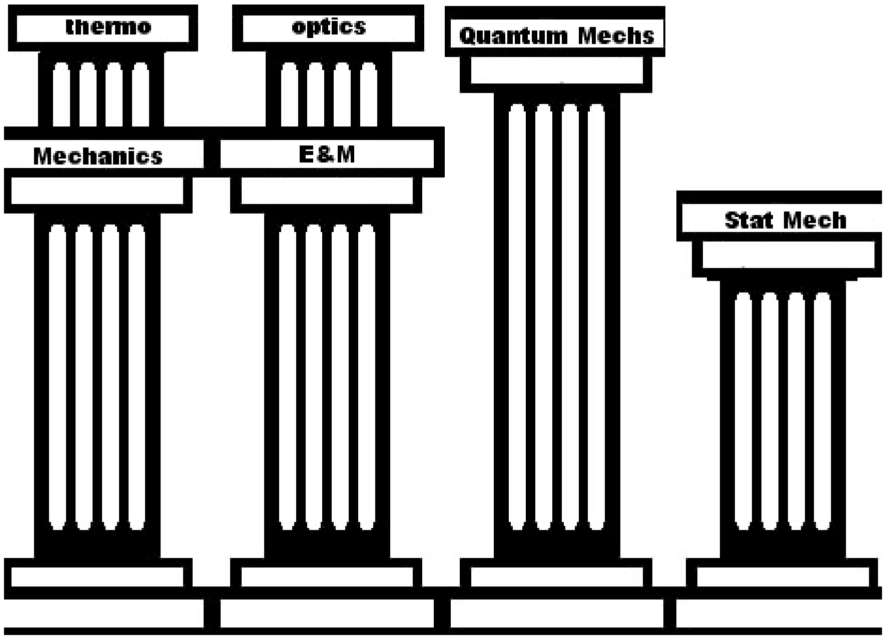
Four pillars:



First time through, math tools are being acquired

Curriculum: the "teach a few subjects and teach them twice" model ...

Four pillars:

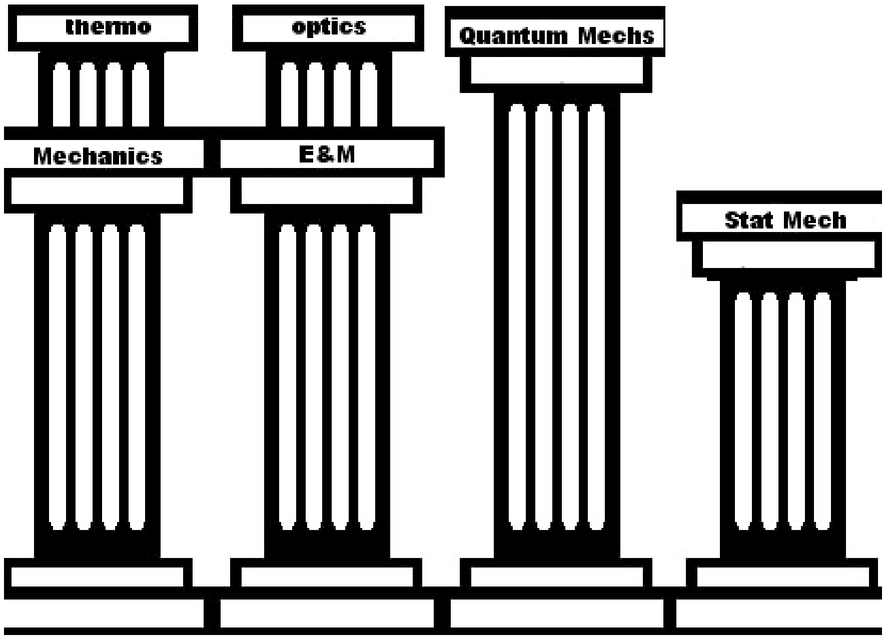


First time through, math tools are being acquired

as are computational tools.

Curriculum: the "teach a few subjects and teach them twice" model …

Four pillars:



First time through, math tools are being acquired

as are computational tools.

Upper-level courses are taught as seminars

Syllabus: Sophomore spring Math methods course

Spring, 1989

- Multivariable and vector calculus
- Complex numbers and analysis
- Diffeq's: ordinary, partial, Frobenius, Green's functions
- Special functions
- Fourier series
- Integral transforms
- Linear algebra
- Calculus of variations
- Probability and statistics
- **Numerical methods** …

- Matrix operations
- Curve fitting
- Error analysis
- Integration
- Fourier analysis
- Root Finding
- Monte Carlo

Sophomore spring
Math methods course

Spring, 1989:
Lecture course

e.g. Root Finding

Spring, 2007:
Lecture course +
Computational Lab taught in both **Matlab** and **Mathematica**
(see D. Cook, 2005 or R. Landau, 2005)

```
DEF FUN(X) = .......
TOL = 1.0E-06
X = 1
DX = .5
FOLD = FUN(X)
ITER = 0
WHILE ABS(DX) > TOL
  ITER = ITER + 1
  X = X+DX
  PRINT ITER, X
  IF FOLD*FUN(X) > 0 THEN GOTO 10
   X = X-DX
   DX = DX/2

10 WEND
STOP
```

```
In[26]:=

fun[x_] := ... ;
FindRoot[fun[x] == 0, {x, -1, 1},
                     Method -> "Brent"]
```

Sophomore spring
Math methods course

Spring, 1989:
Lecture course

e.g. Root Finding

```
DEF FUN(X) = .......
TOL = 1.0E-06
X = 1
DX = .5
FOLD = FUN(X)
ITER = 0
WHILE ABS(DX) > TOL
  ITER = ITER + 1
  X = X+DX
  PRINT ITER, X
  IF FOLD*FUN(X) > 0 THEN GOTO 10
   X = X-DX
   DX = DX/2

10 WEND
STOP
```

- *weekly lab in computer classroom*
- *departmental "living room" has computers / software*
- *campus is wired*
- *keyed software, campus-wide licenses*

Spring, 2007:
Lecture course +
Computational Lab taught in both **Matlab** and **Mathematica**
(see D. Cook, 2005 or R. Landau, 2005)

In[26]:=

```
fun[x_] := ... ;
 FindRoot[fun[x] == 0, {x, -1, 1},
                       Method -> "Brent"]
```

Core Competencies

- I/O
- 2d and 3d visualization
- calculation
- simulation

Core Competencies

- I/O
- 2d and 3d visualization
- calculation
- simulation

Math methods
Lecture course

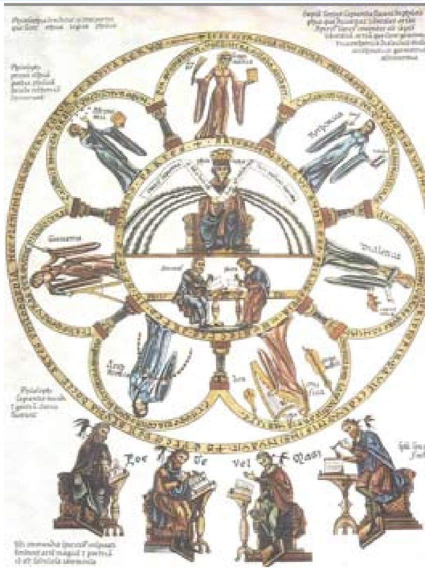
- ✓ Real and complex arithmetic
- ✓ Single and multivariable Calculus
- ✓ Linear Algebra
- ✓ ODEs and PDEs
- ✓ Fourier analysis
- ✓ Special functions
- ✓ Probability/Statistics

Core Competencies

- I/O
- 2d and 3d visualization
- calculation
- simulation

Math methods
Lecture course

- ✓Real and complex arithmetic
- ✓Single and multivariable Calculus
- ✓Linear Algebra
- ✓ODEs and PDEs
- ✓Fourier analysis
- ✓Special functions
- ✓Probability/Statistics



Liberal arts

- ✓Grammar
- ✓Rhetoric
- ✓Logic
- ✓Geometry
- ✓Arithmetic
- ✓Astronomy
- ✓Music

*Curriculum: Weeks 1-7 with **Matlab***

- *Text is home-grown (@author John Boccio) set of notes +exercises.*
- *Read notes*
- *Do several exercises*
- *Emphasis on practical skills, not theory.*

1. **Basics:** GUI and interpreter, getting help, matrix arithmetic, 2d plotting
2. **More basics:** m-files, loops and branching, built-in functions, 3d plotting, image and vector field plotting, reading and writing (formatted and un-) data files
3. **Favorite Algorithms:** derivatives, root-finding, interpolation, definite integration
4. **Monte Carlo (MC):** generating random numbers, MC integration, simple MC simulation (left-right jumping particles in box)
5. **Initial value problems:** different integration algorithms, 1st and 2nd order and systems of ODEs,
6. **Data analysis:** Read file on atmospheric CO_2 vs. time. Fourier transform to remove annual cycle, back transform, do linear regression to find upward trend. Do similar Fourier analysis to find cycles in sunspot data.
7. **Solving PDE's:** shooting, relaxation

Curriculum: Weeks 8-13 with ***Mathematica***

- *Text is Nick Wheeler's set of tutorial notebooks and exercises.*
- *Type and evaluate cells in one or two notebooks*
- *Do several exercises.*
- *Emphasis on practical skills, not theory.*

Physicist's Introduction to *Mathematica*

Physics 200
Fall Semester 2000
Nicholas Wheeler
REED COLLEGE

TABLE OF CONTENTS

*Curriculum: Weeks 8-13 with **Mathematica***

- *Text is Nick Wheeler's set of tutorial notebooks and exercises.*
- *Type and evaluate cells in one or two notebooks*
- *Do several exercises.*
- *Emphasis on practical skills, not theory.*

Unopened notebook

Parametric Plot

```
In[44]:= ? ParametricPlot
```

First Example: Simple Cycloid

A circle of unit radius rolls along the x-axis. We are interest in the curve traced by a point P marked on the circumference of the circle. Working from a sketch, we are led to define

```
In[45]:= x[θ_] := θ - Sin[θ]
         y[θ_] := 1 - Cos[θ]

In[47]:= ParametricPlot[{x[θ], y[θ]}, {θ, 0, 6π}];
```

Curriculum: Weeks 8-13 with *Mathematica*

- *Text is Nick Wheeler's set of tutorial notebooks and problems.*
- *Type and evaluate cells in one or two notebooks*
- *Do several problems.*
- *Emphasis on practical skills, not theory.*

Problems

PROBLEMS

Mathematica Lab Number 1

Problem 1. Evaluate

$$\int_0^\pi \cos\left(x \sin\theta\right) d\theta$$

and use **Plot[%, {x,0,20}];** to plot the famous result. Demonstrate that **Plot[Evaluate[$\int_0^\pi \cos(x\sin\theta)\,d\theta$], {x,0,20}];** does the same job without the distraction of intermediate output.

Problem 2. The Fibonacci numbers are defined recursively

$$F_1 = F_2 = 1 \quad \text{and} \quad F_n = F_{n-1} + F_{n-2} \;:\; n = 3, 4, 5, \ldots$$

and grow very rapidly: ask *Mathematica* about **?Fibonacci** and then evaluate F_{50}. Next construct the generating function

$$\sum_{n=1}^{\infty} \frac{1}{n!} F_n x^n$$

Problem Solutions

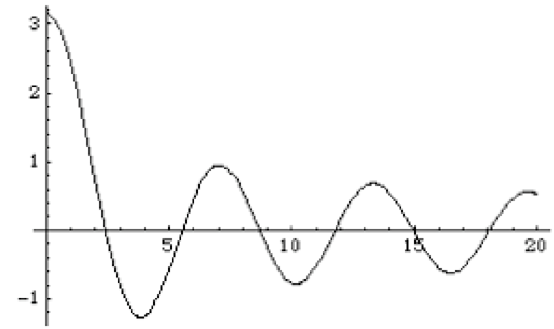
Answers: *Mathematica* Lab 1

■ **Problem 1**

In[2]:= $\int_0^\pi$ **Cos[x Sin[θ]] dθ**

Out[2]= If[Im[x] == 0, π BesselJ[0, x], $\int_0^\pi$ Cos[x Sin[θ]] dθ]

In[3]:= **Plot[%, {x, 0, 20}];**



Why teach students computational tools?

***Most** students **choose to use them** for*

seminar presentations, homework, research, …

Using *Mathematica* to find Clebsch-Gordan Coeffecients

▪ ClebschGordan[{ j_1 , m_1 }, { j_2 , m_2 }, { j, m }] gives the Clebsch-Gordan coefficient for the decomposition of $| j, m\rangle$ in terms of $| j_1, m_1 \rangle\ | j_2, m_2\rangle$.

For example ...

```
ClebschGordan[{2, -2}, {1, -1}, {3, -3}]
```

```
1
```

Why teach students computational tools?

Most *students* ***choose to use them*** *for seminar presentations, homework, research, …*

Find the decomposition of |j,m⟩

```
j1 = 2;
j2 = 1;
j = 3;
m = 2;

(* do not edit below this line *)
a = {}; b = {}; p = "";
For[m2 = j2, m2 ≥ -j2 , m2--,
  If[Abs[m - m2] ≤ j1, a = Join[a, {{m - m2, m2}}]];
 ];
For[i = 1, i ≤ Length[a], i++,
  b = Join[b, {ClebschGordan[{j1, a[[i]][[1]]}, {j2, a[[i]][[2]]}, {j, m}]}]
 ];
For[i = 1, i ≤ Length[b], i++,
  p = StringJoin[p, ToString[If[b[[i]] == 1, "", b[[i]]], StandardForm],
    "|", ToString[j1, StandardForm], ",", ToString[a[[i]][[1]], StandardForm],
    "⟩|", ToString[j2, StandardForm], ",", ToString[a[[i]][[2]], StandardForm],
    "⟩"];
  If[i ≠ Length[b], p = p <> " + "];
 ];
p = StringJoin["|", ToString[j, StandardForm], ",",
   ToString[m, StandardForm], "⟩ = ", p];
Print[p];
```

$$|3,2\rangle = \sqrt{\frac{2}{3}}\,|2,1\rangle|1,1\rangle + \frac{1}{\sqrt{3}}\,|2,2\rangle|1,0\rangle$$

Why teach students computational tools?

***Most** students **choose to use them** for seminar presentations, homework, research, …*

List all Clebsch-Gordan coeffecients

In[1]:=

```
j1 = 3/2;
j2 = 1/2;

(* do not edit below this line *)
Clear[j, m]
(*StylePrint[
   "Valid combinations {m1,m1} such that m1+m2 = m, |m1| ≤ j1, |
     m2| ≤ j2, and corresponding Clebsch-Gordan coeffecients:",
   "Text"];*)
```

• • •

$$|2,2\rangle = \left|\frac{3}{2},\frac{3}{2}\right\rangle\left|\frac{1}{2},\frac{1}{2}\right\rangle$$

$$|2,1\rangle = \frac{\sqrt{3}}{2}\left|\frac{3}{2},\frac{1}{2}\right\rangle\left|\frac{1}{2},\frac{1}{2}\right\rangle + \frac{1}{2}\left|\frac{3}{2},\frac{3}{2}\right\rangle\left|\frac{1}{2},-\frac{1}{2}\right\rangle$$

$$|2,0\rangle = \frac{1}{\sqrt{2}}\left|\frac{3}{2},-\frac{1}{2}\right\rangle\left|\frac{1}{2},\frac{1}{2}\right\rangle + \frac{1}{\sqrt{2}}\left|\frac{3}{2},\frac{1}{2}\right\rangle\left|\frac{1}{2},-\frac{1}{2}\right\rangle$$

$$|2,-1\rangle = \frac{1}{2}\left|\frac{3}{2},-\frac{3}{2}\right\rangle\left|\frac{1}{2},\frac{1}{2}\right\rangle + \frac{\sqrt{3}}{2}\left|\frac{3}{2},-\frac{1}{2}\right\rangle\left|\frac{1}{2},-\frac{1}{2}\right\rangle$$

$$|2,-2\rangle = \left|\frac{3}{2},-\frac{3}{2}\right\rangle\left|\frac{1}{2},-\frac{1}{2}\right\rangle$$

$$|1,1\rangle = -\frac{1}{2}\left|\frac{3}{2},\frac{1}{2}\right\rangle\left|\frac{1}{2},\frac{1}{2}\right\rangle + \frac{\sqrt{3}}{2}\left|\frac{3}{2},\frac{3}{2}\right\rangle\left|\frac{1}{2},-\frac{1}{2}\right\rangle$$

$$|1,0\rangle = -\frac{1}{\sqrt{2}}\left|\frac{3}{2},-\frac{1}{2}\right\rangle\left|\frac{1}{2},\frac{1}{2}\right\rangle + \frac{1}{\sqrt{2}}\left|\frac{3}{2},\frac{1}{2}\right\rangle\left|\frac{1}{2},-\frac{1}{2}\right\rangle$$

$$|1,-1\rangle = -\frac{\sqrt{3}}{2}\left|\frac{3}{2},-\frac{3}{2}\right\rangle\left|\frac{1}{2},\frac{1}{2}\right\rangle + \frac{1}{2}\left|\frac{3}{2},-\frac{1}{2}\right\rangle\left|\frac{1}{2},-\frac{1}{2}\right\rangle$$

Why teach students computational tools?

***Most** students **choose to use them** for*

seminar presentations, homework, research, …

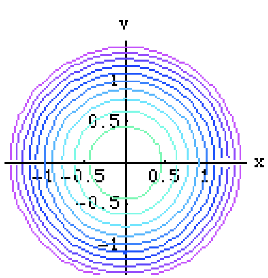
Phase space plots

In[42]:=
```
V[x_] := k*x^2/2;;

k = 1;
m = 1;
e = 0.1;

F = D[V[x], x]

Emin = -1;
Emax = 1;

Xmin = -10;
Xmax = 10;
NCurves = 20;
```

Out[58]= Sin[x]

In[64]:=
```
Off[Plot::plnr];
Plot[Evaluate[{√(2/m*(# - V[x])), -√(2/m*(# - V[x]))} & /@ {Sequence @@ Range[Emin, Emax, (Emax - Emin)/NCurves]}],
  {x, Xmin, Xmax}, PlotStyle → Flatten[Evaluate[{Hue[#], Hue[#]} & /@ {Sequence @@ Range[0, 0.8, 0.8/NCurves]}]],
  PlotPoints → 200, AxesLabel → {x, v}, AspectRatio → Automatic];
```

v

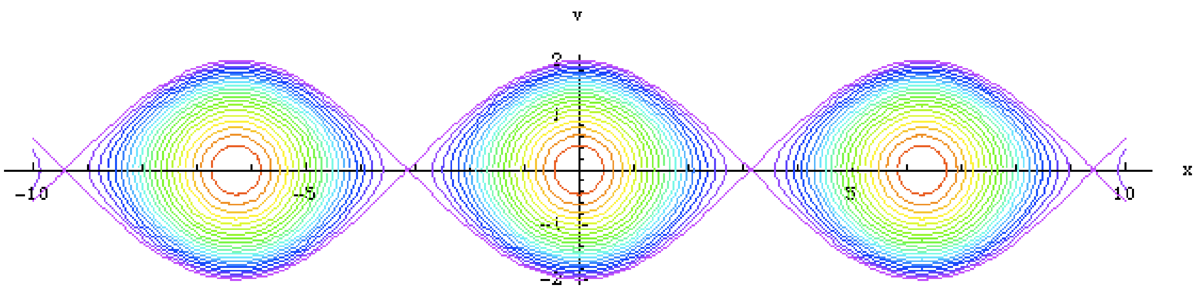


Why teach students computational tools?

Most *students* ***choose to use them*** *for seminar presentations, homework, research, …*

Phase space plots

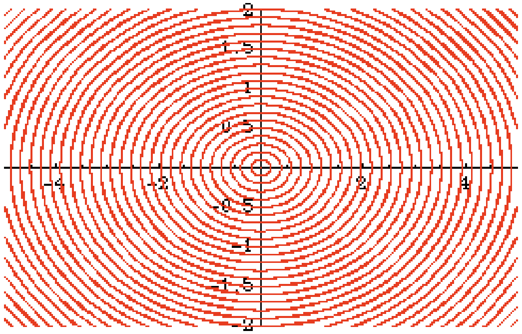
```
In[54]:= V[x_] := -k*Cos[x];

         k = 1;
         m = 1;
         e = 0.1;

         F = D[V[x], x]

         Emin = -1;
         Emax = 1;

         Xmin = -10;
         Xmax = 10;
         NCurves = 20;

Out[58]= Sin[x]

In[64]:= Off[Plot::plnr];
         Plot[Evaluate[{√(2/m*(# - V[x])), -√(2/m*(# - V[x]))} & /@ {Sequence @@ Range[Emin, Emax, (Emax - Emin)/NCurves]}],
           {x, Xmin, Xmax}, PlotStyle → Flatten[Evaluate[{Hue[#], Hue[#]} & /@ {Sequence @@ Range[0, 0.8, 0.8/NCurves]}]],
           PlotPoints → 200, AxesLabel → {x, v}, AspectRatio → Automatic];
```



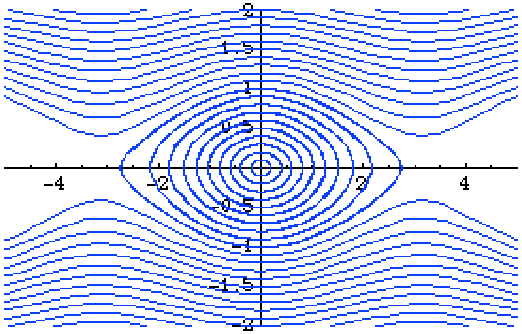
Why teach students computational tools?
***Most** students **choose to use them** for*
seminar presentations, homework, research, …

Phase space plots

```
In[68]:= g1 = ParametricPlot[
    Evaluate[
     {x[t], y[t]} /. NDSolve[{x'[t] == y[t], y'[t] == -0.26 * x[t], x[0] == 0, y[0] == #}, {x[t], y[t]}, {t, -30, 30}][[1]] & /@
      {Sequence @@ Range[-3.5, 3.5, .1]}
    ], {t, -20, 20}, PlotRange → {{-5, 5}, {-2, 2}}, PlotStyle → RGBColor[1, 0, 0]];
```



```
In[69]:= g2 = ParametricPlot[
    Evaluate[
     {x[t], y[t]} /. NDSolve[{x'[t] == y[t], y'[t] == -0.26 * Sin[x[t]], x[0] == 0, y[0] == #}, {x[t], y[t]}, {t, -30, 30}][[1]] & /@
      {Sequence @@ Range[-3.5, 3.5, .1]}
    ], {t, -20, 20}, PlotRange → {{-5, 5}, {-2, 2}}, PlotStyle → RGBColor[0, 0, 1]];
```

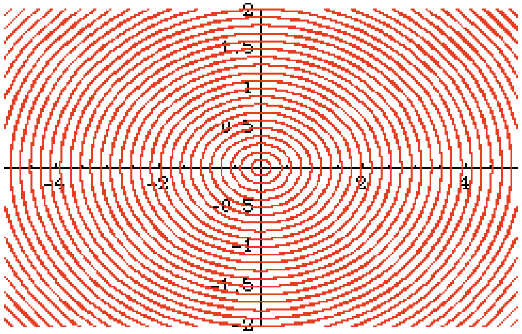


Why teach students computational tools?

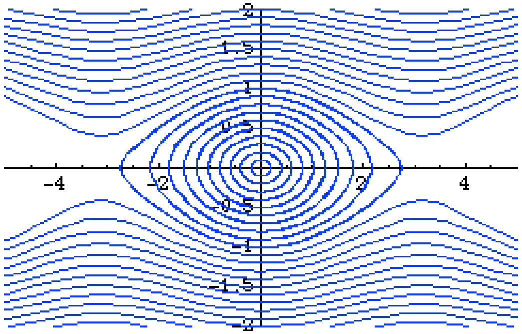
Most *students* ***choose to use them*** *for*
seminar presentations, homework, research, …

Phase space plots

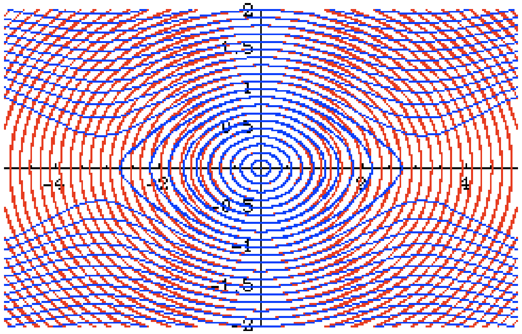
In[68]:=
```
g1 = ParametricPlot[
    Evaluate[
     {x[t], y[t]} /. NDSolve[{x'[t] == y[t], y'[t] == -0.26 * x[t], x[0] == 0, y[0] == #}, {x[t], y[t]}, {t, -30, 30}][[1]] & /@
      {Sequence @@ Range[-3.5, 3.5, .1]}
    ], {t, -20, 20}, PlotRange → {{-5, 5}, {-2, 2}}, PlotStyle → RGBColor[1, 0, 0]];
```



In[69]:=
```
g2 = ParametricPlot[
    Evaluate[
     {x[t], y[t]} /. NDSolve[{x'[t] == y[t], y'[t] == -0.26 * Sin[x[t]], x[0] == 0, y[0] == #}, {x[t], y[t]}, {t, -30, 30}][[1]] & /@
      {Sequence @@ Range[-3.5, 3.5, .1]}
    ], {t, -20, 20}, PlotRange → {{-5, 5}, {-2, 2}}, PlotStyle → RGBColor[0, 0, 1]];
```



In[70]:=
```
Show[g1, g2];
```



Why teach students computational tools?
Most *students* ***choose to use them*** *for*
seminar presentations, homework, research, …

```
(*tom's and ben's amazing problem*)

<< Calculus`VectorAnalysis`
SetCoordinates[Spherical[R, θ, ϕ]];

x = {Sin[θ] * Cos[ϕ], Cos[θ] * Cos[ϕ], -Sin[ϕ]};
y = {Sin[ϕ] * Sin[θ], Cos[θ] * Sin[ϕ], Cos[ϕ]};
p = Cos[ω * t] * x + Sin[ω * t] * y;
r = {1, 0, 0};

Efield = -η * Cross[r, Cross[r, p]] // FullSimplify
Bfield = β * Cross[r, p] // FullSimplify
S = 1 / μ * Cross[Efield, Bfield] // FullSimplify
```

$$\left\{0,\ \frac{\text{p0}\,\mu\,\omega^2\,\text{Cos}[\theta]\,\text{Cos}[\phi - t\,\omega]}{4\,\pi\,R},\ -\frac{\text{p0}\,\mu\,\omega^2\,\text{Sin}[\phi - t\,\omega]}{4\,\pi\,R}\right\}$$

$$\left\{0,\ \frac{\text{p0}\,\mu\,\omega^2\,\text{Sin}[\phi - t\,\omega]}{4\,c\,\pi\,R},\ \frac{\text{p0}\,\mu\,\omega^2\,\text{Cos}[\theta]\,\text{Cos}[\phi - t\,\omega]}{4\,c\,\pi\,R}\right\}$$

$$\left\{\frac{\text{p0}^2\,\mu\,\omega^4\,(\text{Cos}[\theta]^2\,\text{Cos}[\phi - t\,\omega]^2 + \text{Sin}[\phi - t\,\omega]^2)}{16\,c\,\pi^2\,R^2},\ 0,\ 0\right\}$$

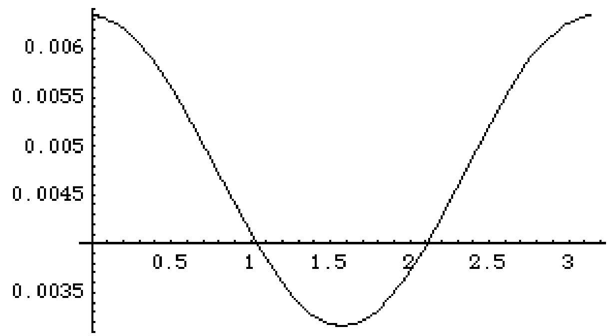
```
β = μ * p0 * ω^2 / (4 π * c * R);
η = μ * p0 * ω^2 / (4 π * R);
```

$$\text{Savg} = \frac{\text{p0}^2\,\mu\,\omega^4\,(\text{Cos}[\theta]^2 * 1/2 + 1/2)}{16\,c\,\pi^2\,R^2}\ \text{// FullSimplify}$$

```
ans = Integrate[Integrate[Savg * R^2 * Sin[θ], {ϕ, 0, 2 π}],
  {θ, 0, π}]

Plot[Savg /. μ → 1 /. p0 → 1 /. c → 1 /. R → 1 /. ω → 1, {θ, 0, π},
  PlotRange → Automatic];
```

$$\frac{\text{p0}^2\,\mu\,\omega^4\,(3 + \text{Cos}[2\,\theta])}{64\,c\,\pi^2\,R^2}$$

$$\frac{\text{p0}^2\,\mu\,\omega^4}{6\,c\,\pi}$$



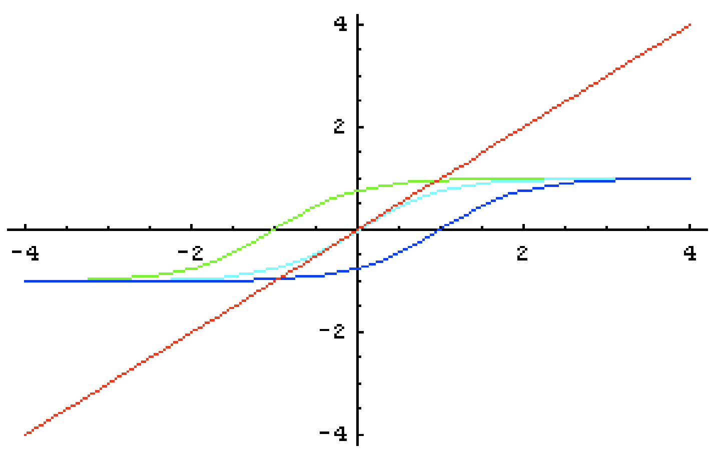
Why teach students computational tools?

***Most** students **choose to use them** for*

seminar presentations, homework, research, …

Schroeder chapter 8 prob 22.nb

```
Eup = -ϵ * n * sbar - μ * B;
Edown = ϵ * n * sbar + μ * B;

Zi = e^(β * Eup) + e^(β * Edown) // FullSimplify

2 Cosh[β (n sbar ϵ + B μ)]

savg = 1 / Zi * (-1 * e^(β * Eup) + 1 * e^(β * Edown)) //
  FullSimplify

Tanh[β (n sbar ϵ + B μ)]

saverage[A_, x_, β_, B_] := Tanh[β (A x + B)]

Plot[{saverage[1, x, 1, 1], saverage[1, x, 1, 0],
   saverage[1, x, 1, -1], x}, {x, -4, 4},
  PlotStyle → {RGBColor[0, 1, 0], RGBColor[0, 1, 1],
    RGBColor[0, 0, 1], RGBColor[1, 0, 0]}]
```



Why teach students computational tools?

***Most** students **choose to use them** for*
seminar presentations, homework, research, …

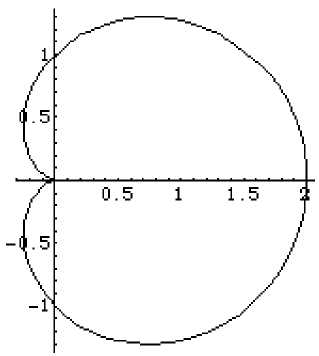
FINAL EXAM

#1 Consider the central force orbit $r = a(1+\cos\theta)$

(a) Sketch the orbit

(b) Find the form of the central force that produced it

```
<< Graphics`

r[t_] := a (1 + Cos[t]);

PolarPlot[r[t] /. a → 1, {t, 0, 2 π}];
```




```
Force = F /. Solve[D[1 / r[t], {t, 2}] + 1 / r[t] == -μ * r[t]^2 / l^2 * F, F] //
   FullSimplify // First
```

$$-\frac{3\, l^2}{a^3\, \mu\, (1 + \mathrm{Cos}[t])^4}$$

```
Potential = -Integrate[-3 * a * l^2 / μ * 1 / R^4, R] // FullSimplify
```

$$-\frac{a\, l^2}{R^3\, \mu}$$

Math methods Curriculum: What's missing?

What we'd cover in a real Computational Physics course (seminar) …

CS Basics …

- OS
- a high-level compiled language (modular, oop-capable)
- edit > compile/link > execute
- IDE
- good coding habits (whitespace, commenting, error handling)

CP Basics … But now "theory"
as well as hands-on skill acquisition

Individualized Projects

Topics we don't do but I wish we could …

- *debugging environment*
- *performance profiling*
- *code maintenance*
- *Finite element methods*
- *HPC and parallel programming*

Home page: A CP Seminar

Senior Honors Study

Physics 199, Spring 1999

Computational Physics

Seminarians: B.Huff, J. Lifton, W. Luh, S. Lukin, J. Pyle, A. Bug (Prof)

Main text by: Landau and Paez. **Additional texts by:** Gould and Tobochnik, Pang, Gibbs, Haile, Press et al., Allen and Tildesley, Giordano.

This seminar will introduce computer calculations as a way of solving problems in various fields of physics: classical mechanics, electricity and magnetism, quantum mechanics, statistical and chemical physics... We are going to learn concepts of robust scientific computing and explore various techniques like the numerical solution of ODEs, PDEs and eigenvalue problems, Monte Carlo, and FFTs. We are also going to prepare for the external exams that form the culmination of the honors experience at Swarthmore!

If you hope to find detailed seminar assignments at this Website, I'm afraid you'll be disappointed (the Prof still draws the occasional figure and sometimes even writes derivations with a pen! So they get xeroxed and aren't 100% electronic). She would be happy to mail the hard copies to interested readers, who can email her at abug1@swarthmore.edu.

What we *will* do at this Website is chronicle the main events each week:

- Week 1: Introduction to numerical calculations
- Week 2: Errors and Integrals

Home page: Another CP Seminar

Seminar on Computational Physics
Prof. Amy Bug
Spring 2006

Home | **Course Information** | Syllabus | Sample journal page |

- Week 1 introduction
- Week 2 procedural java 1
- Week 3 procedural java 2
- Week 4 objective java
- Week 5 integrating motion
- Week 6 oscillations&orbits
- Week 7 chaos
- Week 8 molecular dynamics
- Week 9 electrodynamics
- Week 10 ditto
- Week 11 MC, stochastic simulations and phase transitions
- Week 12 ditto
- Week 13 project planning
- Week 14 quantum systems

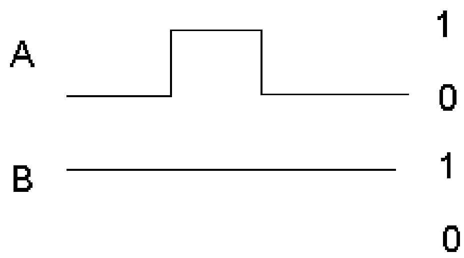
Course materials

Electronic readings
Codes from Amy
Student Codes
Student Work

Links offsite

Open Source Physics
Java I/O Notes

comp phys in the news ... / who's bringing break? / colloquium this Friday /



Course Information

Instructor
Prof. Amy Bug
x8257
SC 117/117a
abug1@swarthmore.edu

Text
An Introduction to Computer Simulation Methods, 3rd Edition (2006) by Gould, Tobochnik and Christian

Meeting place and time
Tuesday evening 7-10PM
Computer classroom, SC256
(Additional hours to work in that classroom, TBA)

CP Seminar Curriculum

Assignment 1: A taste of Computational Physics

Goals: The goals for this week are to acquire basic tools that will be useful throughout the semester, and to gain a glimpse into the field of computational physics. When you finish this seminar, it is hoped that you will be able to ...

- find your way around the Unix operating system (as implemented under Mac OS X) with basic shell commands
- edit, compile, and run simple Java codes from the command line
- "bundle" a code, so that it can be run as a standalone Mac application
- write a simple Java applet that can be run from a WWW Browser (or the appletviewer application)
- be able to edit and run Java source codes using the Eclipse interactive development environment (IDE)
- gain new insights into
  - how one approaches problem-solving with computers
  - how computation has contributed to physics
  - the Java language and programming environment

CP Seminar Curriculum

Assignment 2: Arithmetic, Algorithms and Error

Procedural Java I

Goals: When you use a digital computer to do an applied math or physics calculation, there are basic issues to confront involving numerical precision, choice of algorithm, and error. The goal for this week is to explore these issues, and in doing so, use basic elements of the Java language When you finish this seminar, it is hoped that you will be familiar with

- different digital formats used to represent numbers on the computer
- rounding and truncation errors
- the difference between the order of accuracy of an algorithm, and its stability
- algorithms for root-finding and interpolation
- the following (plus or minus a few) aspects of the Java language:

white space ⋆ creating blocks with braces ⋆ data types ⋆ arithmetic ⋆ assignment and operators (binary, and one ternary) ⋆ initializing data ⋆ scope of data ⋆ if ⋆ while ⋆ for ⋆ using methods in the `Math` class ⋆ 1D arrays

CP Seminar Curriculum

Assignment 3: I/O and Methods

Procedural Java II

Goals: Input/output of information is trivial in principle, but in practice it can be a headache. The more strongly-typed the language, the more inflexible the grammar to get numbers in and out correctly. One goal for this week is to be sure that you can use Java to read and write to both the screen and files. (By next week, we will cultivate the habit of entering data into a graphical user interface as well.) Also this week, you will be studying methods, which in another language would be called functions (if they return a value) or subroutines (if they do not). A method compartmentalizes a single task (e.g. the taking of an integral, the initialization of parameters for a simulation, ...)

When you finish this seminar, it is hoped that you will be familiar with

- ways to format your data so that it is read or written as the correct data type
- ways to do I/O from the command-line
- ways to do I/O from files
- how to create a static method and how to call it from somewhere in your code
- algorithms for differentiation and integration
- (your Phys 50 memories of) the Euler method of integrating an ordinary diffeq
- the following (plus or minus a few) aspects of the Java language:
  static methods (object methods are optional for now, though we will dwell on them starting next week ...) ⋆ passing variables to and from methods ⋆ scope of variables and methods ⋆method overloading ⋆ the dot operator for accessing object variables or methods ⋆ formats for floating point numbers ⋆ the System object and its input, output and error streams ⋆ casts and (widening or narrowing) conversions⋆ how to convert strings to primitive data types ⋆ command line arguments ⋆ buffered readers and writers for doing screen and file I/O

Text?

Homegrown notes (@author: Amy Bug)

Numerical Notes 4
Integration

A taste of Eclipse ...
Documentation written: *December, 2005*
Author: *Amy Bug, Swarthmore College*
Tested on: *Mac OS-X 10.3 or 10.4 ... sorry, no guarantees for other systems ...*

JavaNotes 1:
What is Java and how do I run it?

Textbook : Landau and Paez (1999), GT&C (2006)

Additional textbooks and journal articles: numerous!

Many additional internet-based programming and scientific resources

Assigned work and in-Seminar Activities?

Short exercises: "Everyone problems" … not as useful as I hoped … hard to discuss productively in seminar and assess afterward.

Exercises: Individualized, each week … presentations emphasize physics as well as code … interesting, beneficial, difficulty with viewing each-other's code

Weekly journal: Documents mini-project and other learning … valuable but time-consuming for student…

Projects: Cumulative … interesting, beneficial

Assessment?

Short exercises: "Everyone problems" …

Exercises: *Individualized, each week … presentations emphasize physics as well as code …*

Weekly journal: *Documents mini-project and other learning …*

Projects: *Cumulative …*

Marking criteria are individualized. CS majors and n00bs are not held to the same standard. A's are commonplace.

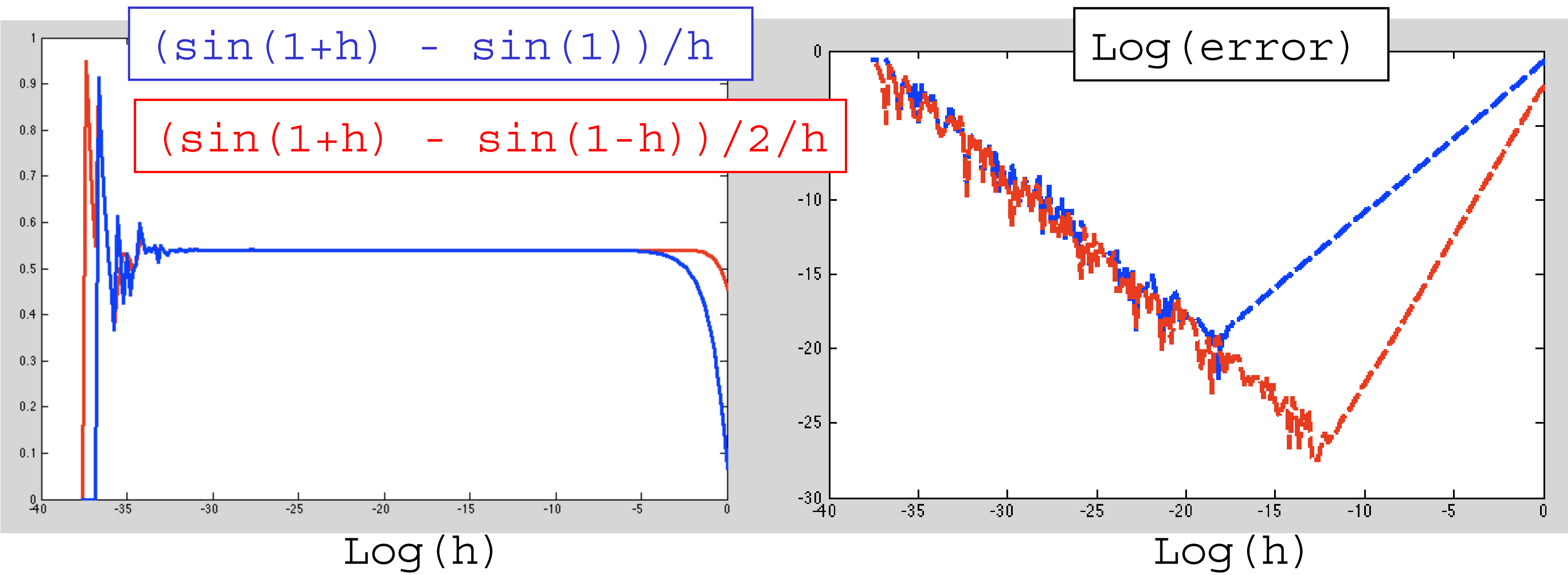
Favorite topics

Cross compare various algorithms ->
Error? Stability?

Favorite topics

Cross compare various algorithms ->
Error? Stability?

Differentiation

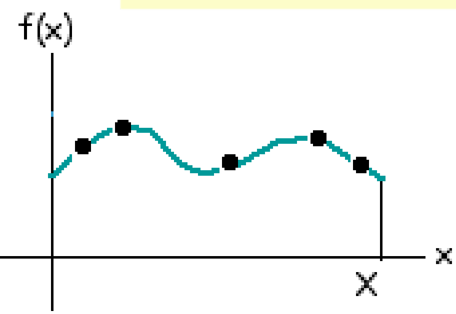


Favorite topics

Cross compare various algorithms ->
Error? Stability?

Integration

"sample mean" method



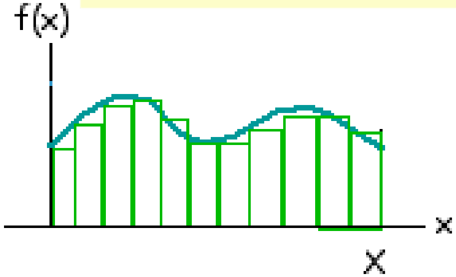
$$I = \int_0^X f(x)dx \; = \; X < f >$$

$$\approx X \frac{1}{N} \sum_{i=1}^{N} f(x_i) \; \pm \; X \, \sigma_f / \sqrt{N}$$

where

$$\sigma_f^2 = < f^2 > - < f >^2$$

as opposed to quadriture



$$I \approx \sum_{i=1}^{N} f(x_i) \Delta x_i \pm ord(N^{-a/D})$$

Favorite topics

Stability of integrator:
Dying vs. growing exponential vs. the Verhulst Eq

$$\frac{dP}{dt} = rP$$

$$\frac{dP}{dt} = rP\left(1 - \frac{P}{K}\right)$$

$$P(t) = \frac{KP_0e^{rt}}{K + P_0\left(e^{rt} - 1\right)}$$

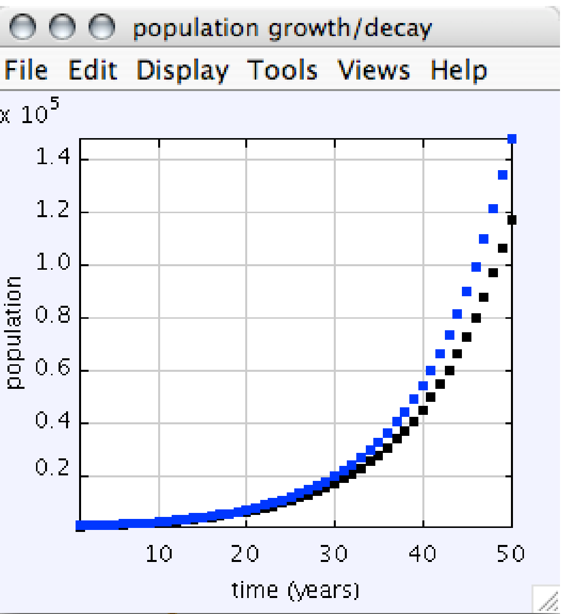
$$\lim_{t\to\infty} P(t) = K$$

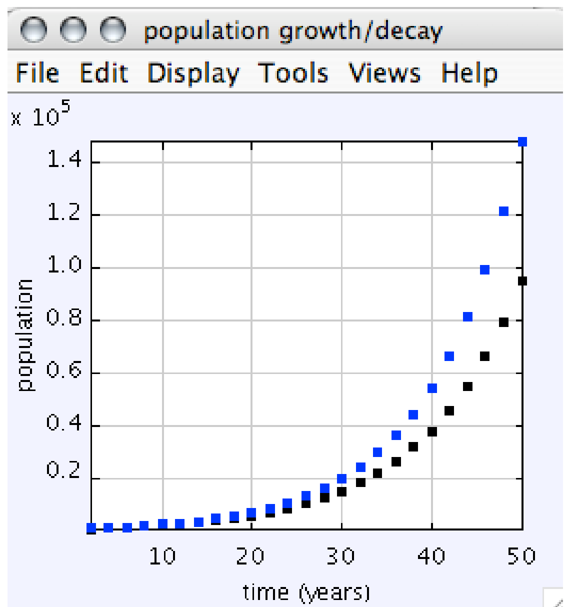
Favorite topics

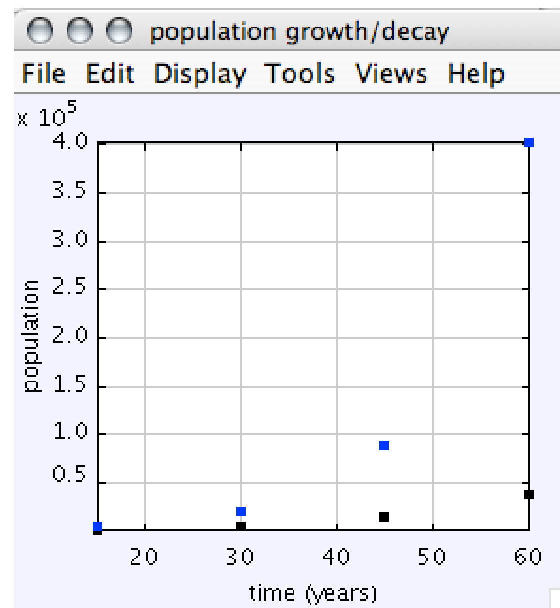
Stability of integrator:
Dying vs. growing exponential vs. the Verhulst Eq

$$\frac{dP}{dt} = rP$$

r > 0: Euler, h increases,





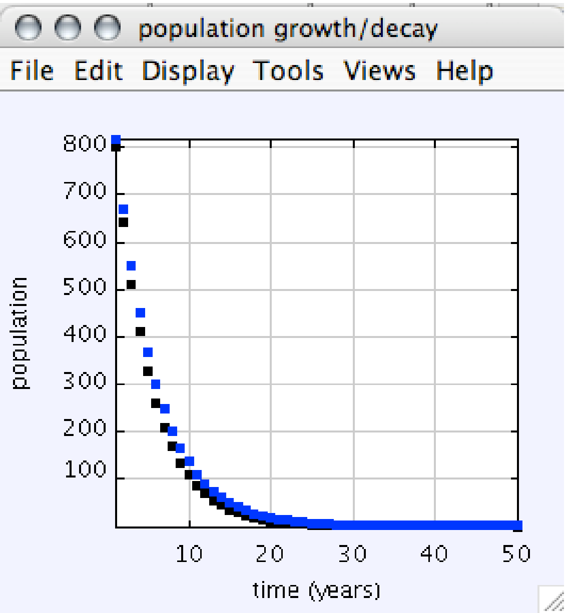


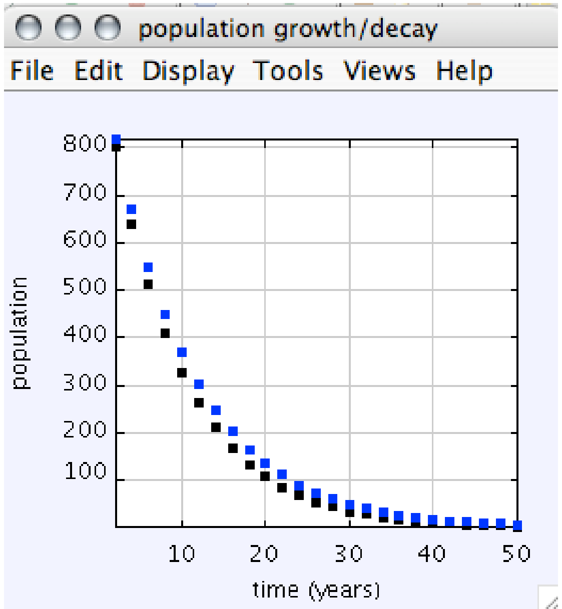
Favorite topics

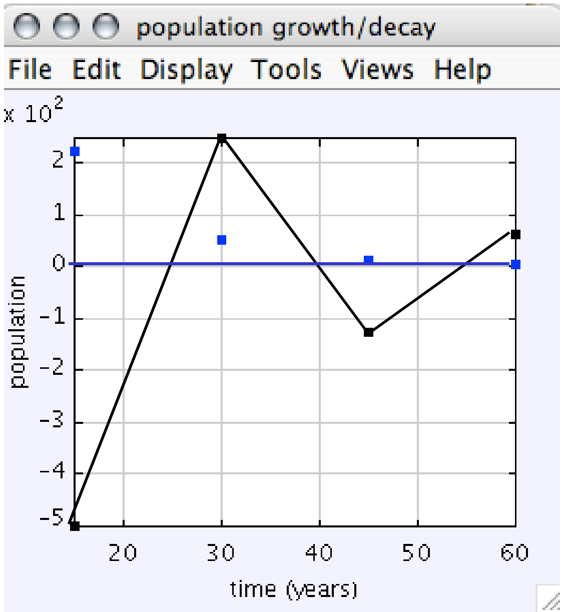
Stability of integrator:
Dying vs. growing exponential vs. the Verhulst Eq

$$\frac{dP}{dt} = rP$$

r < 0: Euler, h increases,







Favorite topics

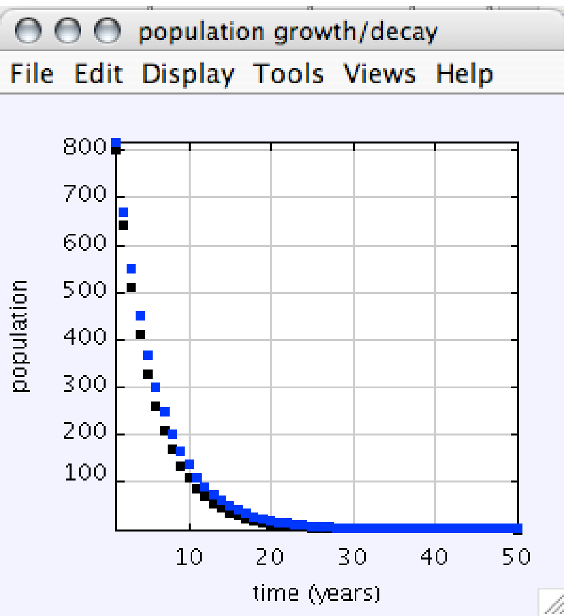
Stability of integrator:
Dying vs. growing exponential vs. the Verhulst Eq

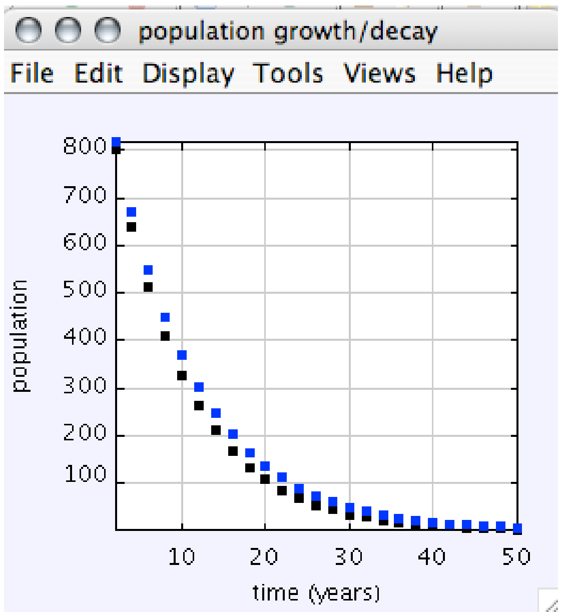
Stability determined by considering propagation of integration error:

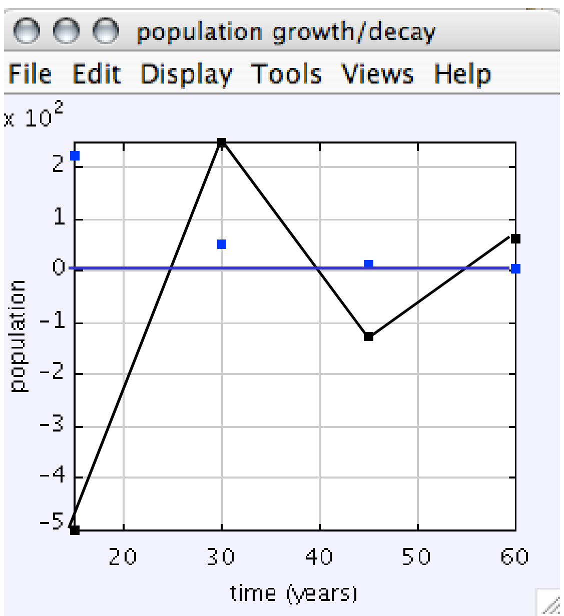
$$e(x+h) \approx e(x)(1+hr)$$

Explicit Euler

$$\frac{dP}{dt} = rP$$





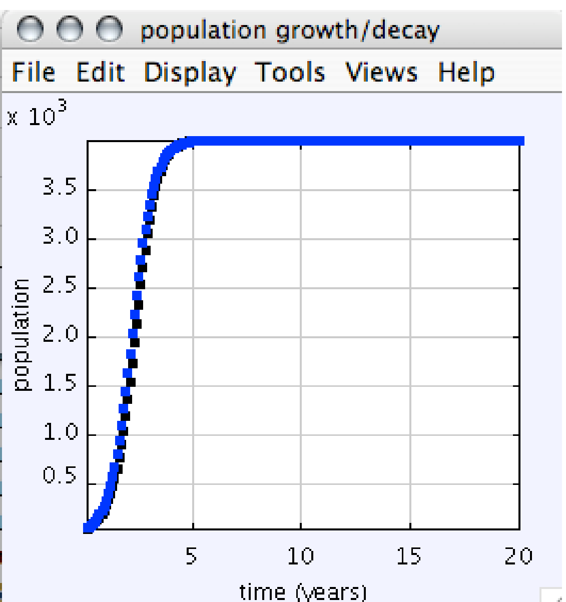


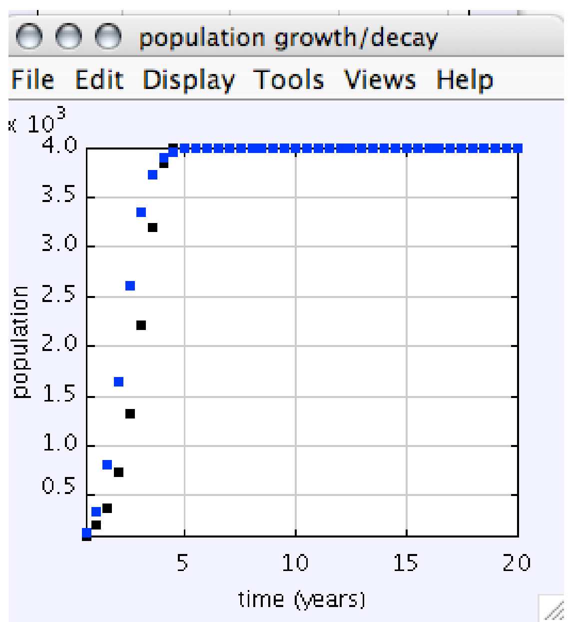
Favorite topics

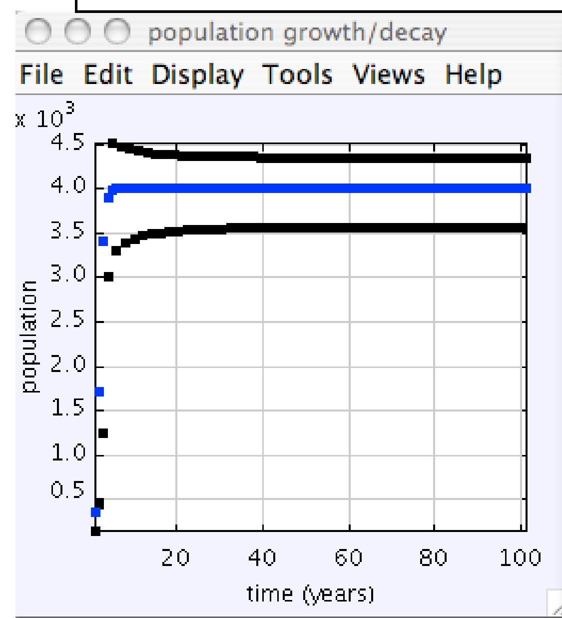
Stability of integrator:
Dying vs. growing exponential vs. the Verhulst Eq

r > 0: Euler, h increases, →

$$\frac{dP}{dt} = rP\left(1 - \frac{P}{K}\right)$$





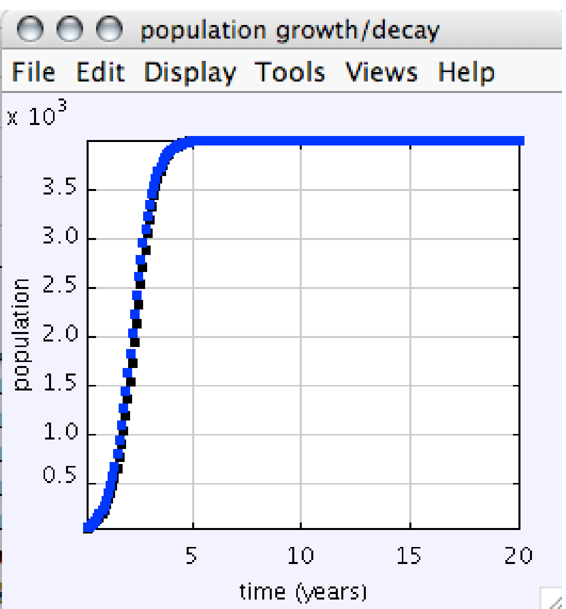


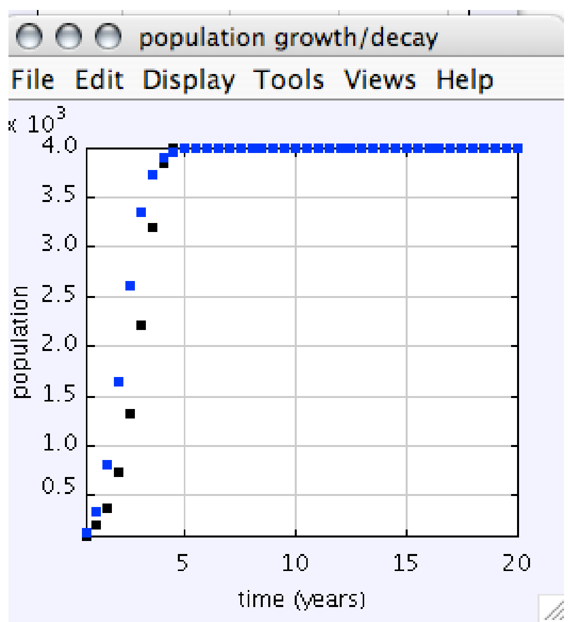
Favorite topics

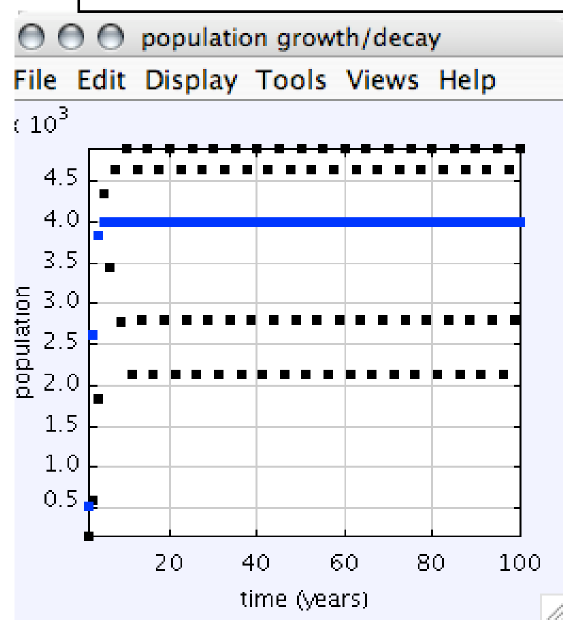
Stability of integrator:
Dying vs. growing exponential vs. the Verhulst Eq

r > 0: Euler, h increases,

$$\frac{dP}{dt} = rP\left(1 - \frac{P}{K}\right)$$





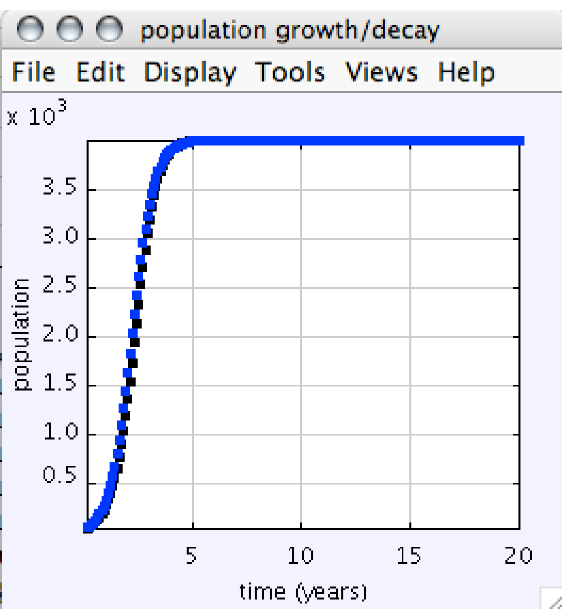


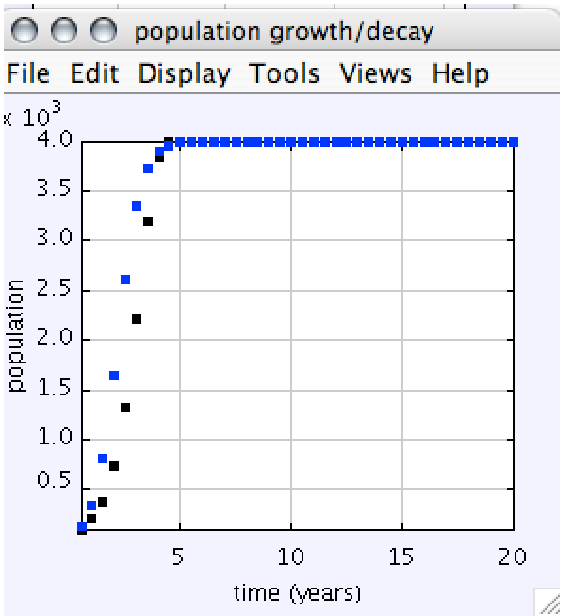
Favorite topics

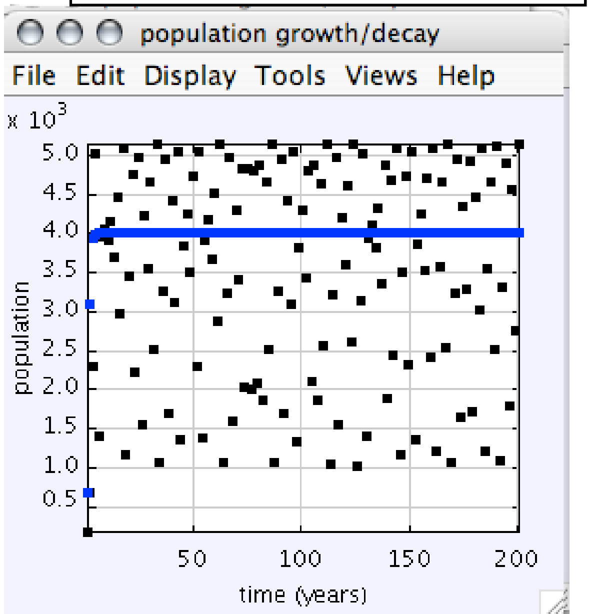
Stability of integrator:
Dying vs. growing exponential vs. the Verhulst Eq

r > 0: Euler, h increases,

$$\frac{dP}{dt} = rP\left(1 - \frac{P}{K}\right)$$





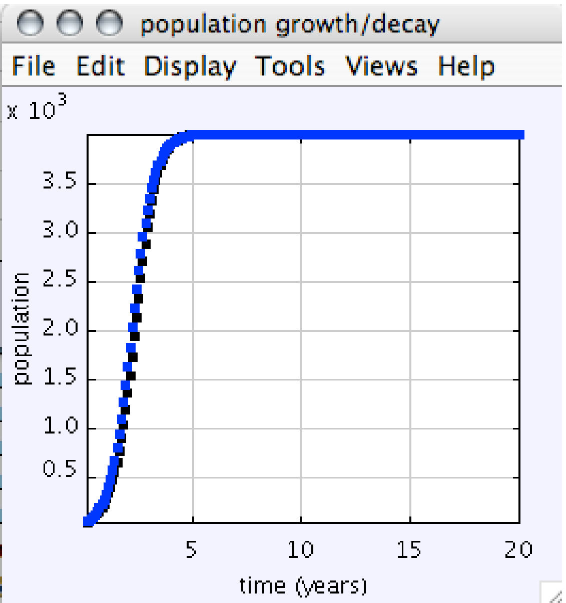


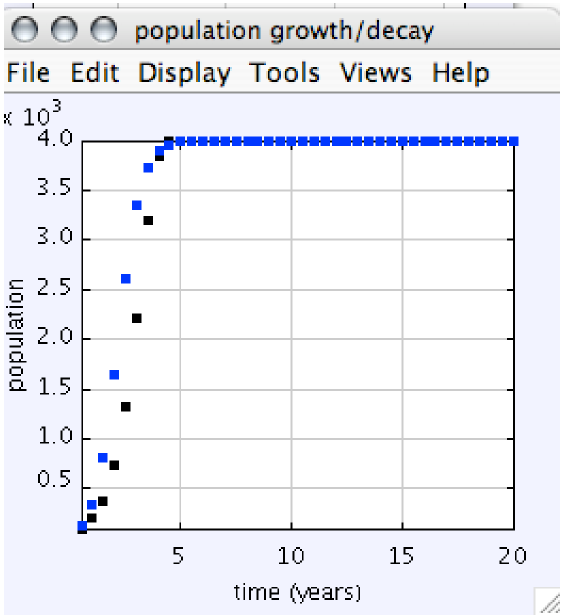
Favorite topics

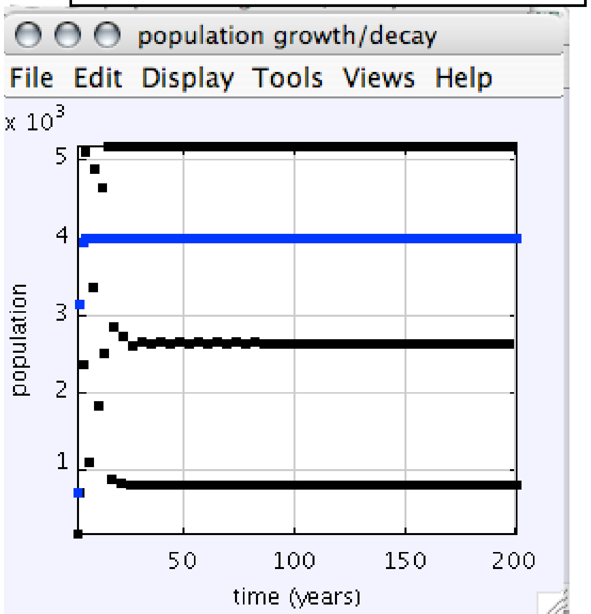
Stability of integrator:
Dying vs. growing exponential vs. the Verhulst Eq

r > 0: Euler, h increases,

$$\frac{dP}{dt} = rP\left(1 - \frac{P}{K}\right)$$







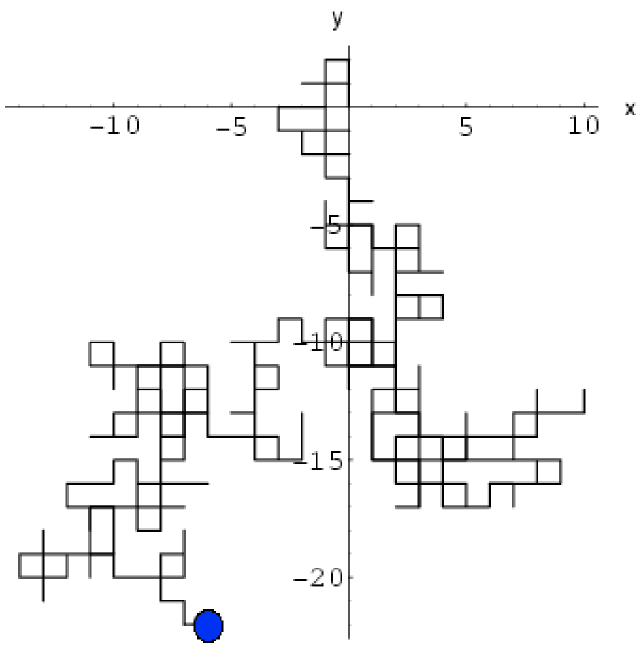
Favorite topics

Same application, very different algorithm … or vica-versa … ->
the unity of physics

Favorite topics

Same application, very different algorithm … or vica-versa … ->
the unity of physics

Iterative finite-element (Relaxation) cf.
Random walk (RW) solution of
Laplace's Equation



$$p(x,y,t+\Delta t) =$$

$$\frac{1}{4}[p(x_+,y,t)+p(x_-,y,t)+p(x,y_+,t)+p(x,y_-,t)]$$

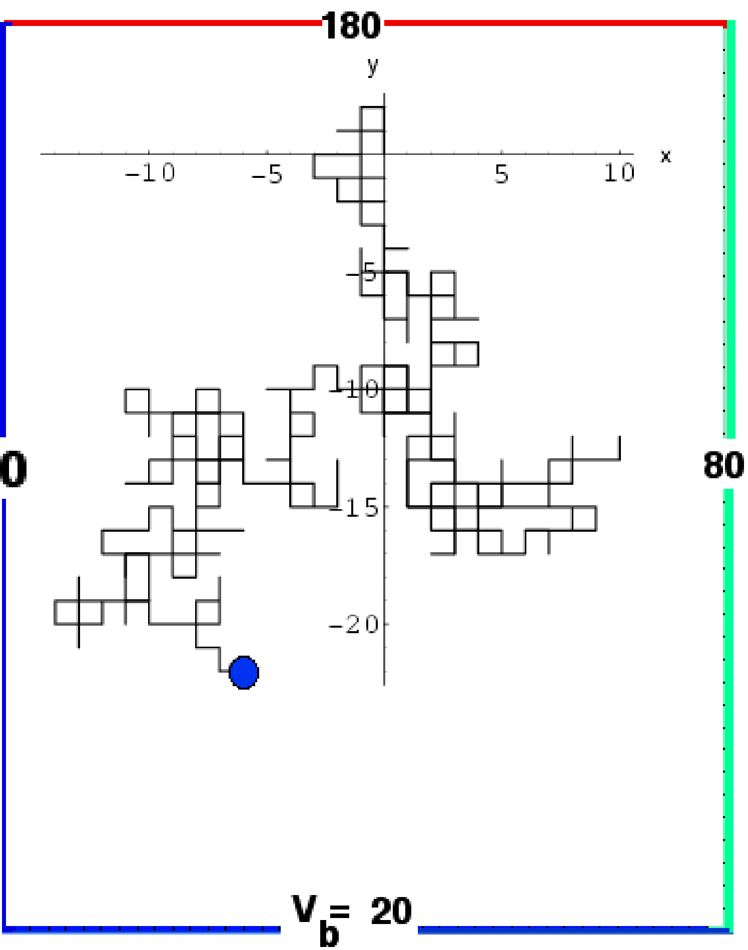
<=> *Master equation, a discretized version of the*

Diffusion equation: $\partial p/\partial t = \frac{a^2}{4\Delta t}\nabla^2 p$ **D**

Steady state DE: $\nabla^2 p = 0$

Laplace's equation: $\nabla^2 V = 0$

Relaxation (matrix methods)
cf.
Random walk (RW) solution of
Laplace Equation

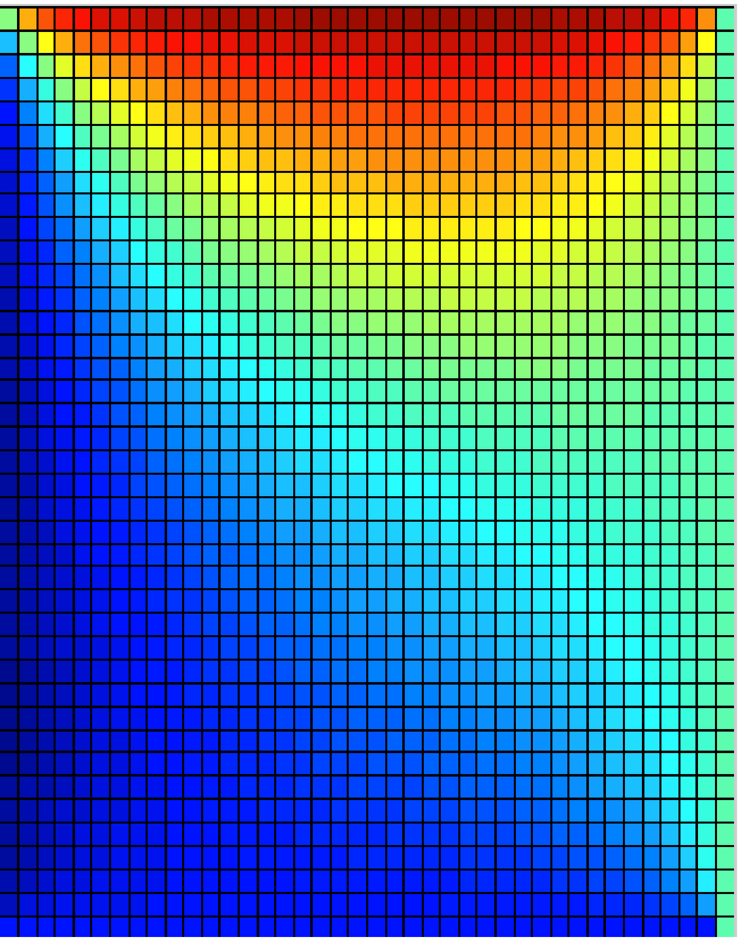


RW algorithm:

1. Begin walker at point (x,y) where V(x,y) is desired.
2. Take random steps until a boundary is reached.
3. Record V_b, the potential at that boundary.
4. Repeat N times ... accumulate $\{V_{b,i}\}$
5. Estimator is

$$V(x,y) \approx \frac{1}{N}\sum_{i=1}^{N} V_{b,i}$$

Relaxation (matrix methods) cf. Random walk (RW) solution of Laplace's Equation



RW algorithm:

1. Begin walker at point (x,y) where V(x,y) is desired.
2. Take random steps until a boundary is reached.
3. Record V_b, the potential at that boundary.
4. Repeat N times ... accumulate $\{V_{b,i}\}$
5. Estimator is

$$V(x,y) \approx \frac{1}{N}\sum_{i=1}^{N} V_{b,i}$$

Relaxation:

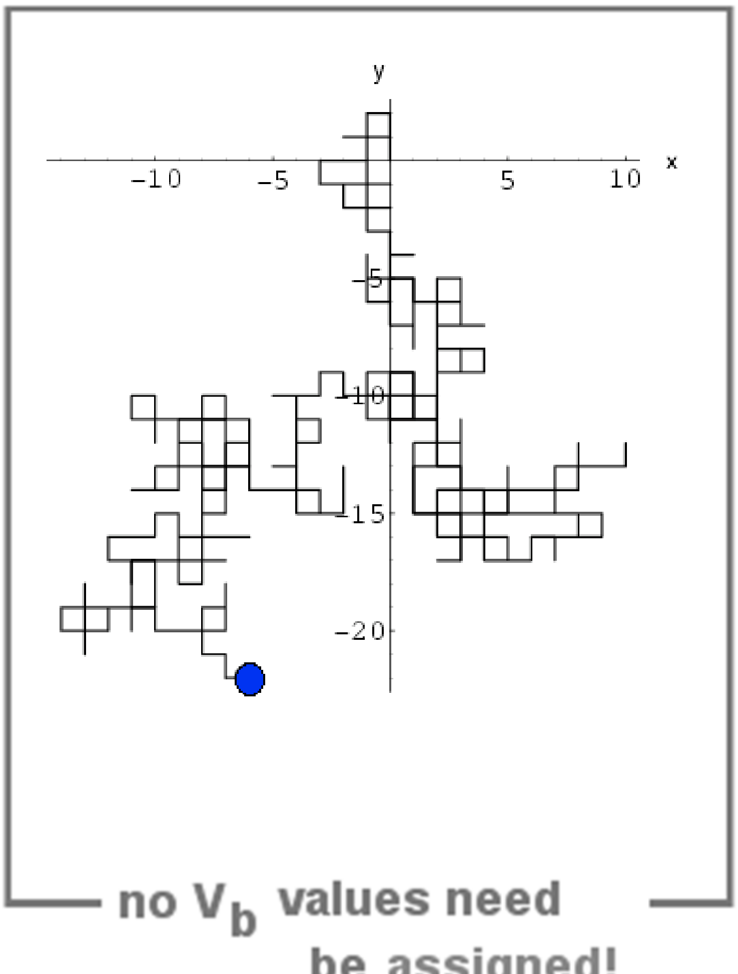
To solve $\mathbf{A}V(x_i,y_j) = 0$, let

$$V(x_i,y_j)_{n+1} = \mathbf{G}_{ijkl}\, V(x_k,y_l)_n$$

e.g. say $\mathbf{A} = \mathbf{M} - \mathbf{N}$

then $\mathbf{G} = \mathbf{M}^{-1}\,\mathbf{N}$

Relaxation (matrix methods)
cf.
Random walk (RW) solution of
Laplace Equation



RW algorithm:

1. Begin walker at point (x,y) where V(x,y) is desired.
2. Take random steps until a boundary is reached.
3. Record V_b, the potential at that

In fact, do not even need to know values on boundary during simulation. Boundary shape is enough. Algorithm finds Green's Function for ∇^2

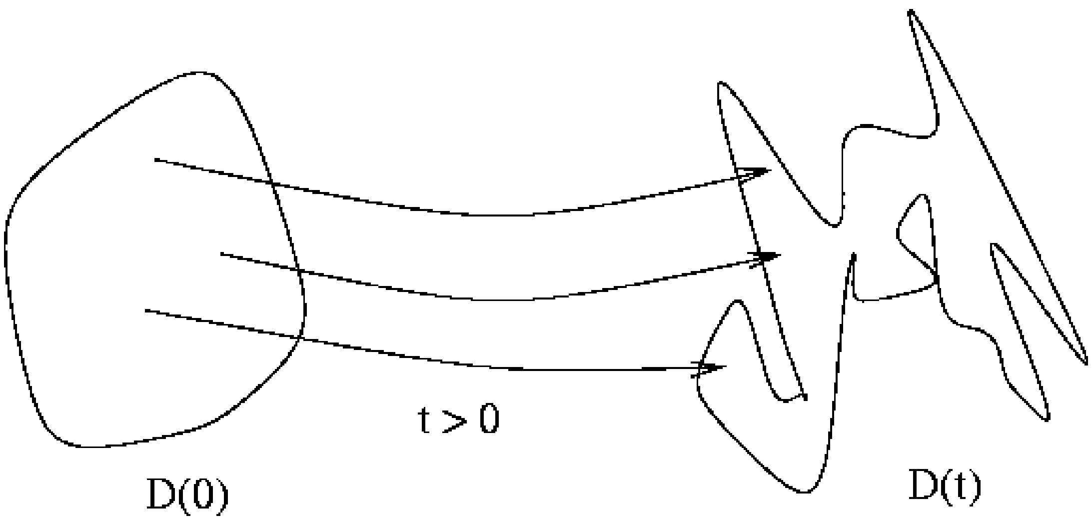
$$V(x,y) = \frac{1}{N_b} \int G(x,y,x_b,y_b) V(x_b,y_b)$$

Favorite topics

Interplay between physics and numerical analysis

Symplectic integrators

Hamilton's Equations imply the Liouville Equation



$$\dot{q}_i = \frac{\partial H}{\partial p_i} \ , \quad \dot{p}_i = -\frac{\partial H}{\partial q_i}$$

$$\frac{\partial \rho(z,t)}{\partial t} + \dot{z} \cdot \nabla_z \rho(z,t) = 0$$

with $z \equiv (q,p)$

Favorite topics

Interplay between physics and numerical analysis

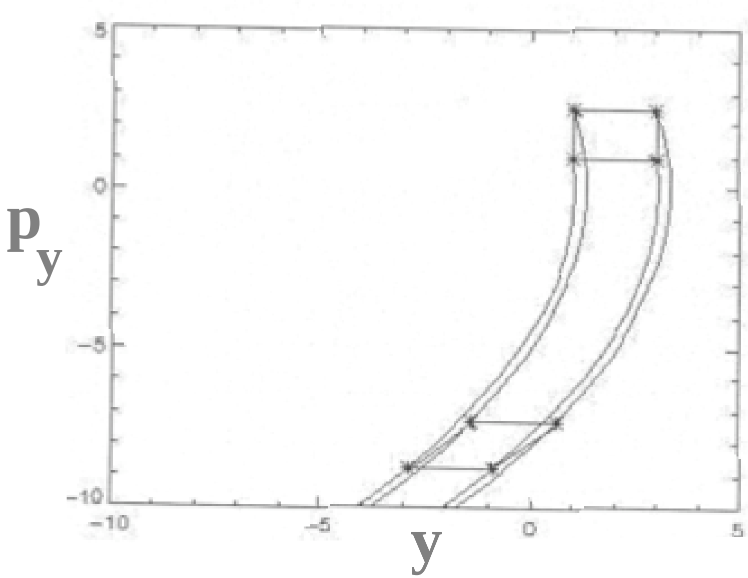
Symplectic integrators

Hamilton's Equations imply a map which evolves (p(t), q(t)) forward in time.

$$\dot{q}_i = \frac{\partial H}{\partial p_i} \; , \quad \dot{p}_i = -\frac{\partial H}{\partial q_i}$$

$$\Leftrightarrow \quad \frac{d}{dt} z = J \, \nabla H(z).$$

with $z \equiv (q, p)$ *and* $J = \begin{pmatrix} 0 & I_n \\ -I_n & 0 \end{pmatrix}$



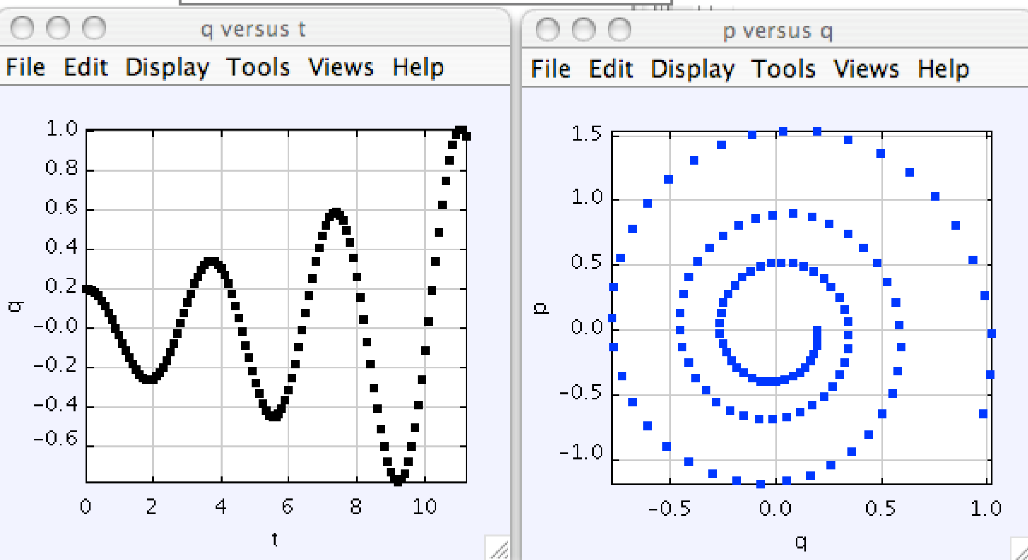
Evolution is symplectic if $M^T J M = J$

where M is matrix of partials of map

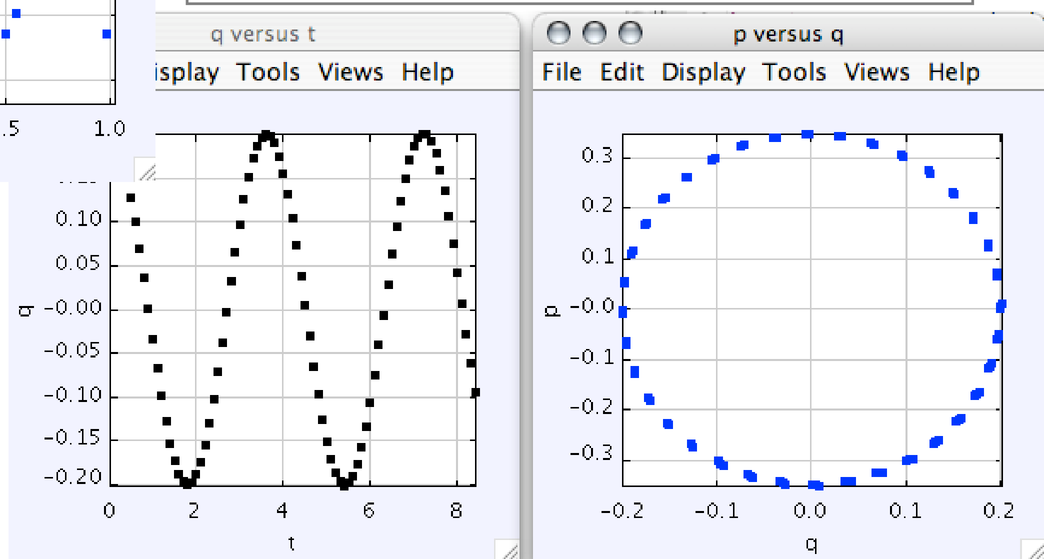
Symplectic integrators

When solving for Hamiltonian dynamics, it is good to use a numerical integrator which is also symplectic …*

*No: **Euler**, RK, …*



*Yes: **Euler-Cromer**, Verlet, …*

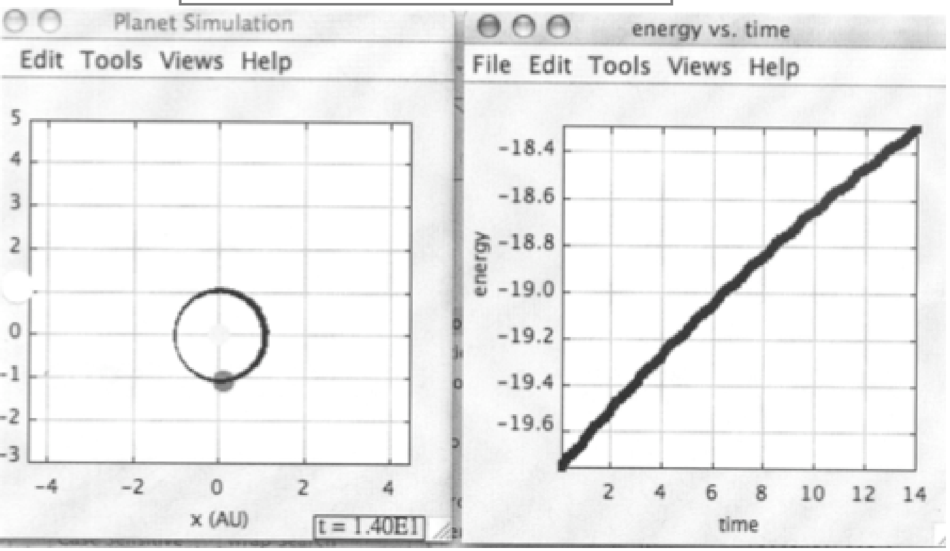


** …up to the expected order of the integrator*

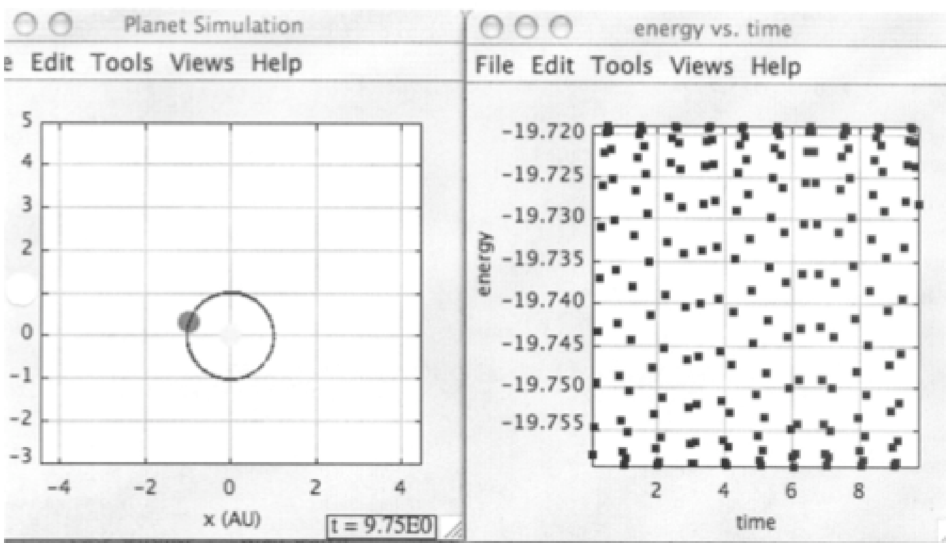
Symplectic integrators

When solving for Hamiltonian dynamics, it is good to use a numerical integrator which is also symplectic …*

*No: Euler, **RK3**, …*



*Yes: Euler-Cromer, **Verlet**, …*

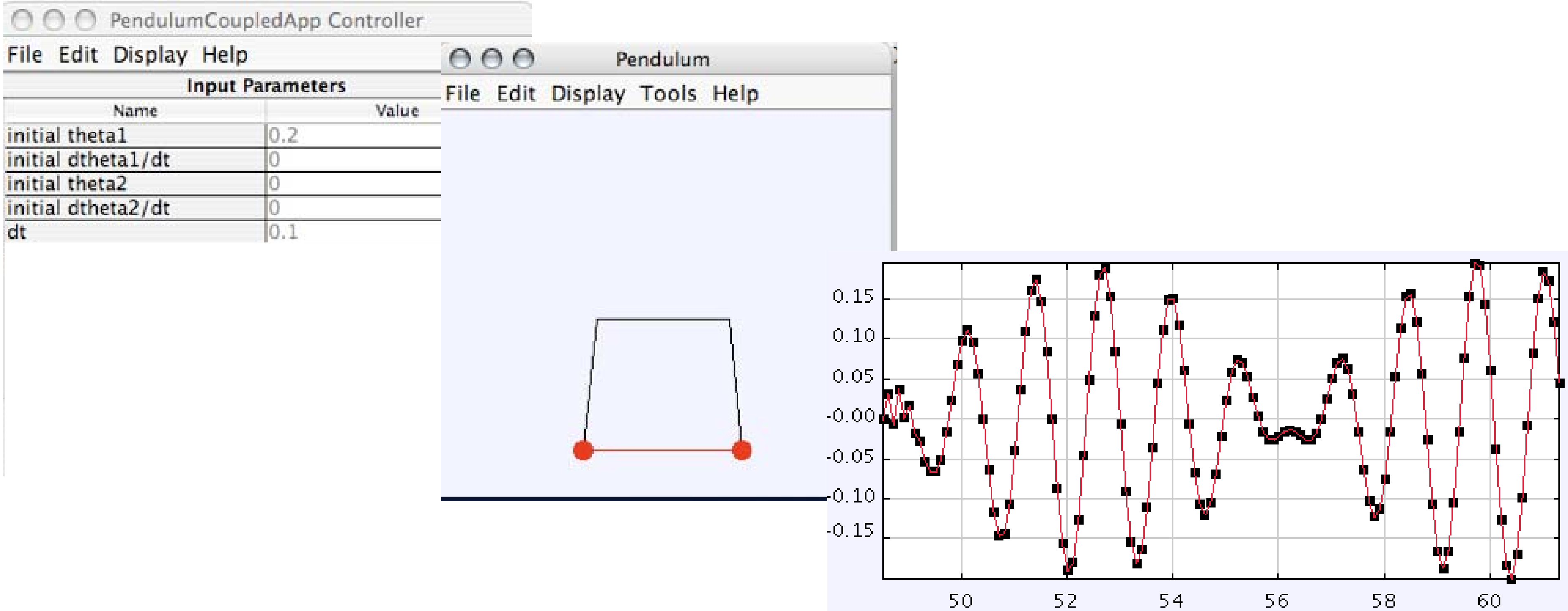


** …up to the expected order of the integrator*

Favorite topics

Using helpful tools

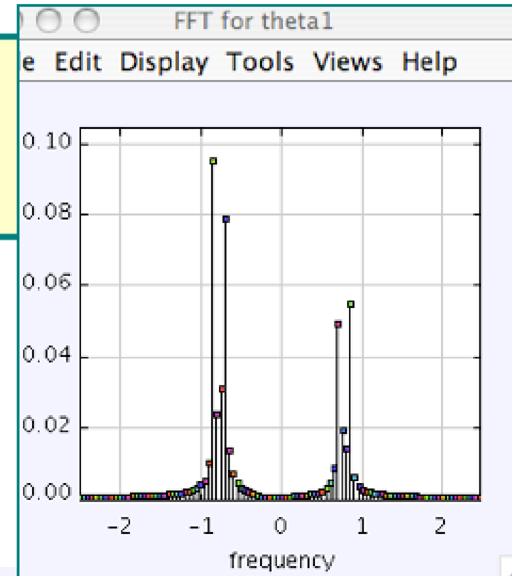
Fourier transforms

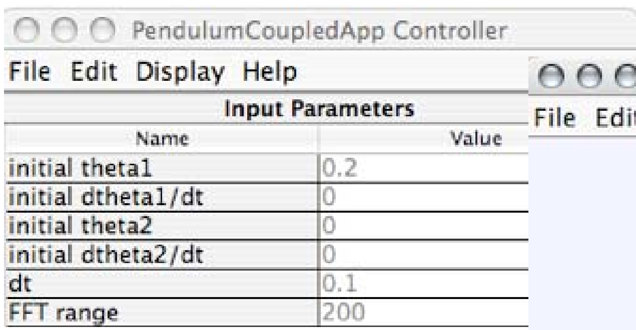


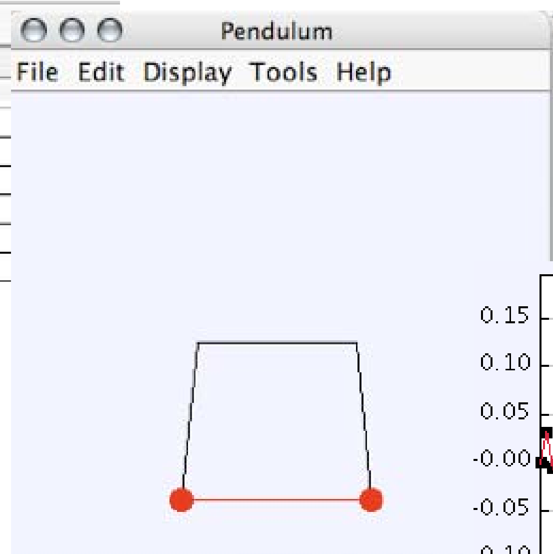
Favorite topics

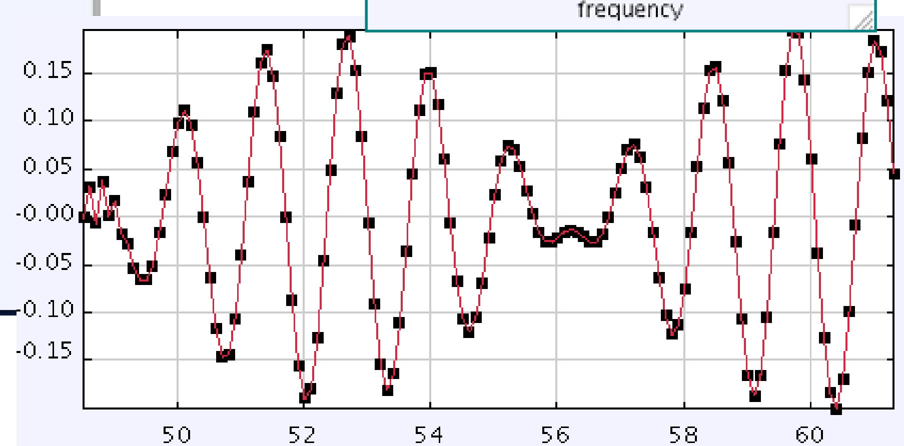
Using helpful tools

Fourier transforms









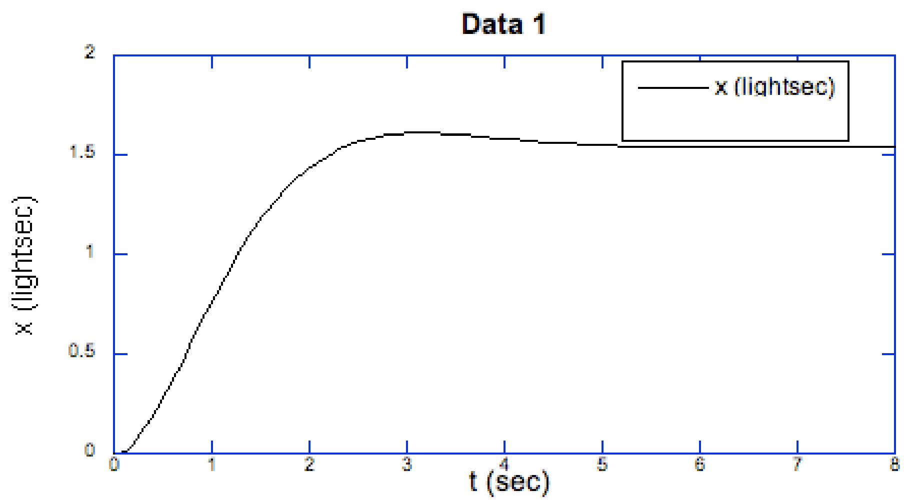
Favorite topics

Dealing with data

Tachyon: Students are given file with t_i and $x(t_i)$

Tachyon.txt

0.0000	0.0000
0.10000	0.013962
0.20000	0.053064
0.30000	0.11204
0.40000	0.18622
0.50000	0.27151
0.60000	0.36435
0.70000	0.46169
0.80000	0.56096
0.90000	0.66000
1.0000	0.75707
1.1000	0.85076
1.2000	0.94000
1.3000	1.0240
1.4000	1.1021
1.5000	1.1741
1.6000	1.2397
1.7000	1.2989
1.8000	1.3518
1.9000	1.3985
2.0000	1.4394
2.1000	1.4748
2.2000	1.5049
2.3000	1.5303
2.4000	1.5514



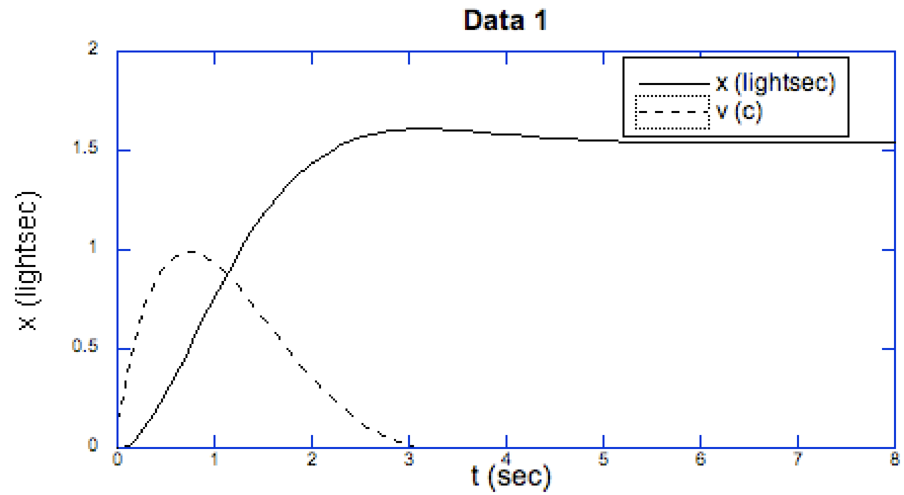
Favorite topics

Dealing with data

Tachyon: Students differentiate $x(t_i)$ … Does $v(t_i)$ ever exceed c ?

Tachyon.txt

0.0000	0.0000
0.10000	0.013962
0.20000	0.053064
0.30000	0.11204
0.40000	0.18622
0.50000	0.27151
0.60000	0.36435
0.70000	0.46169
0.80000	0.56096
0.90000	0.66000
1.0000	0.75707
1.1000	0.85076
1.2000	0.94000
1.3000	1.0240
1.4000	1.1021
1.5000	1.1741
1.6000	1.2397
1.7000	1.2989
1.8000	1.3518
1.9000	1.3985
2.0000	1.4394
2.1000	1.4748
2.2000	1.5049
2.3000	1.5303
2.4000	1.5514



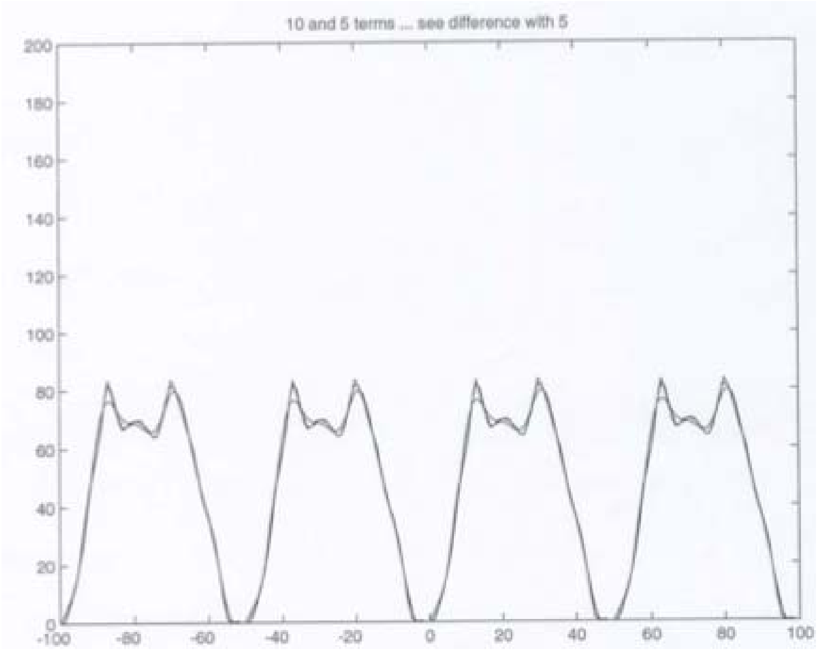
Output for Tachyon.java:

```
Please enter the pathname of the file containing the Tachyon information:
/home/jenny/Tachyon.txt
Please enter the pathname of the file containing the Tachyon analytical data
/home/jenny/Tachyon_soln.txt
Derivative: red, Analytical: blue
red: forward difference
blue: center difference
green: five point
```

Favorite topics

Dealing with data

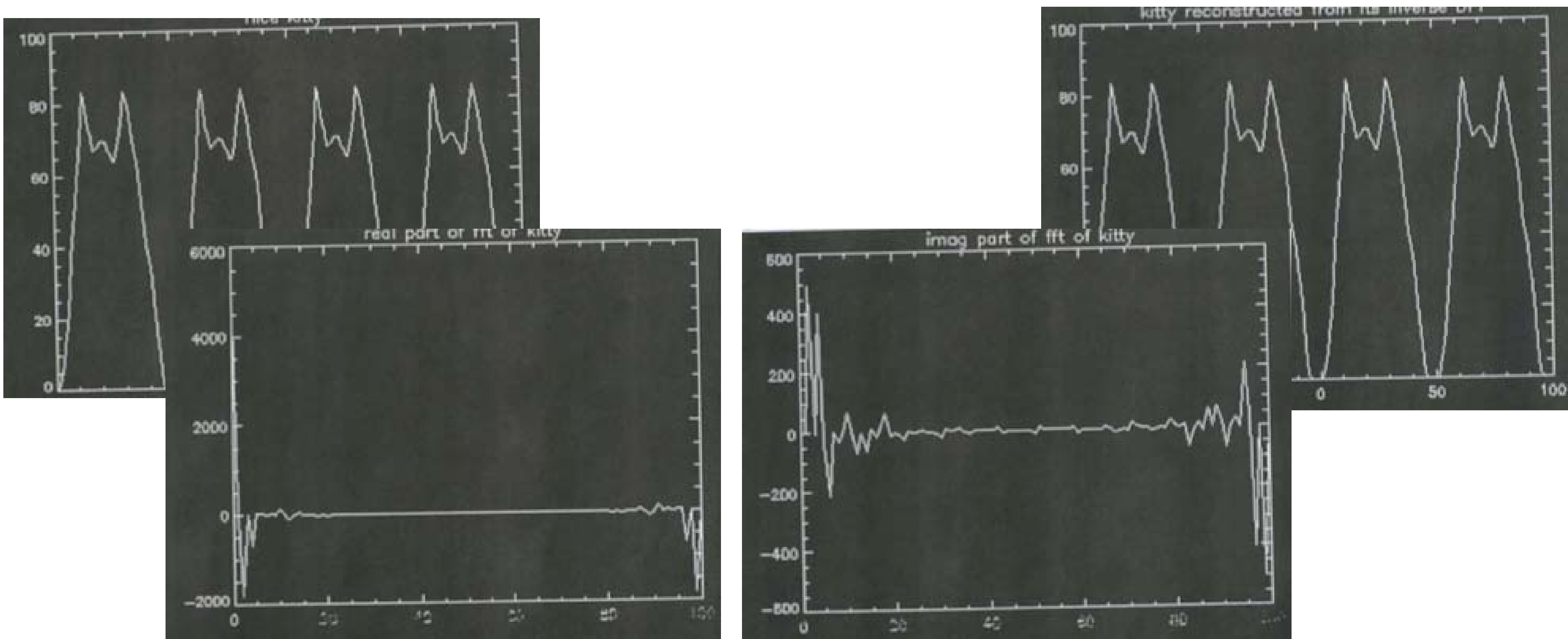
Kitty: Students write a code to find terms of the Fourier series for periodic data, then re-synthesize the original data from it.



Favorite topics

Dealing with data

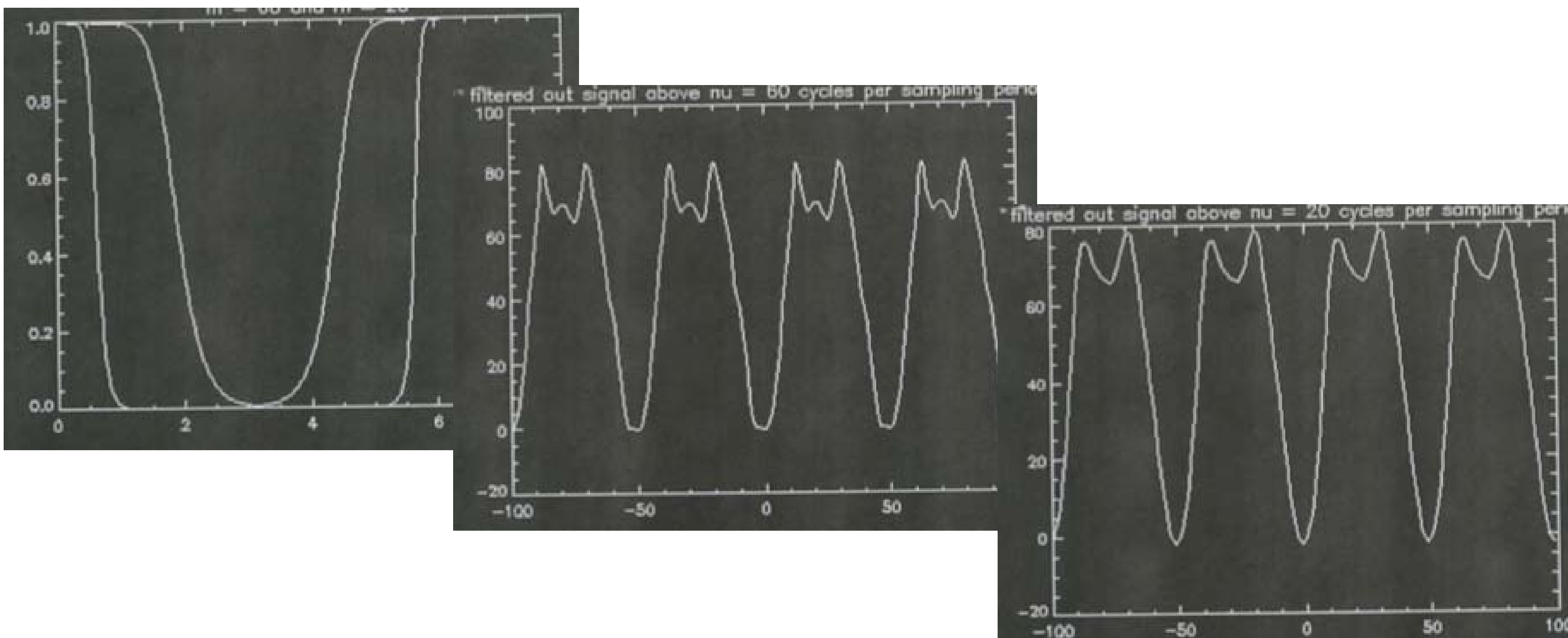
Kitty: Students do a discrete Fourier transform of the data.
They invert it to reconstruct the time series.



Favorite topics

Dealing with data

Kitty: Students write low pass digital filters and apply them to data.



Favorite topics

More than one numerical task ->
Calculation of physical interest

Scattering cross section (Pang's book)

$$\theta = \pi - 2\int_{r_m}^{\infty} \frac{b\,dr}{r^2\sqrt{1 - \frac{b^2}{r^2} - \frac{U(r)}{E}}}$$

$$\text{where } 1 - \frac{b^2}{r_m^2} - \frac{U(r_m)}{E} = 0$$

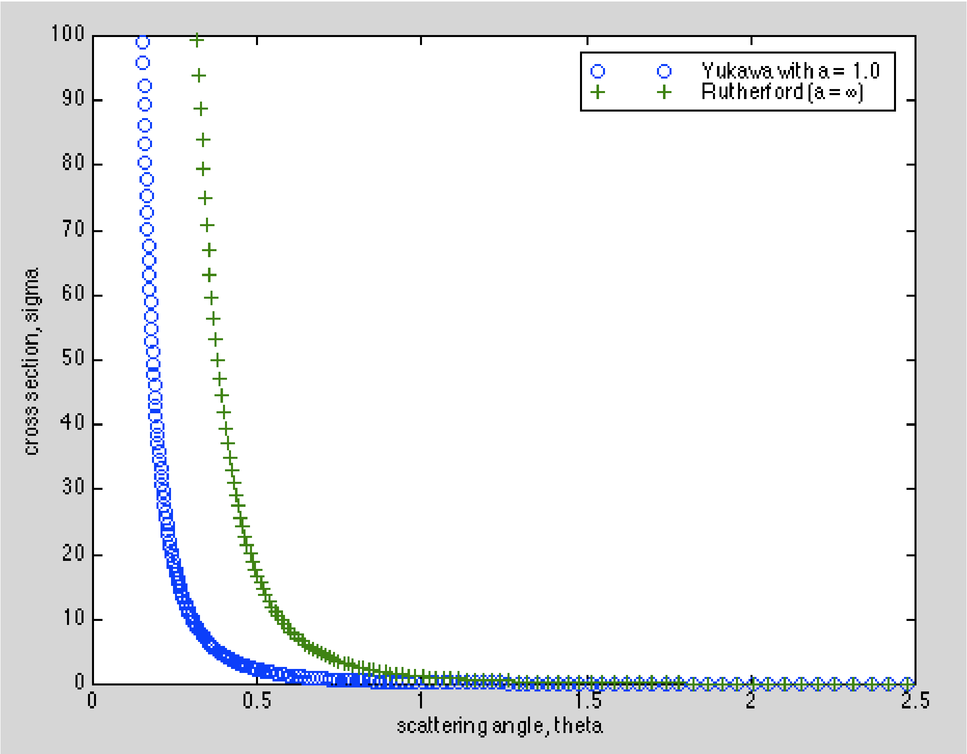
$$\sigma(\theta) = \frac{b}{\sin\theta}\left|\frac{db}{d\theta}\right|$$

- ☐ *Root finding for classical turning point*
- ☐ *Numerical integration for scattering angle, $\theta(b)$*
- ☐ *Interpolation for smooth $\theta(b)$ curve*
- ☐ *Numerical differentiation For $\sigma(\theta)$*

Favorite topics

More than one numerical task ->
Calculation of physical interest

Scattering cross section (Pang's book)



- *Root finding for classical turning point*
- *Numerical integration for scattering angle, $\theta(b)$*
- *Interpolation for smooth $\theta(b)$ curve*
- *Numerical differentiation For $\sigma(\theta)$*

Favorite topics

More than one numerical task ->
Calculation of physical interest

A dynamical simulation

1999 seminar: Coding “round robin"

We used the Velocity Verlet algorithm, LJ pairwise potential,and periodic boundary conditions to simulate a gas of LJ particles in thermal equilibrium.

Within a week's time, we had a working MD simulation!

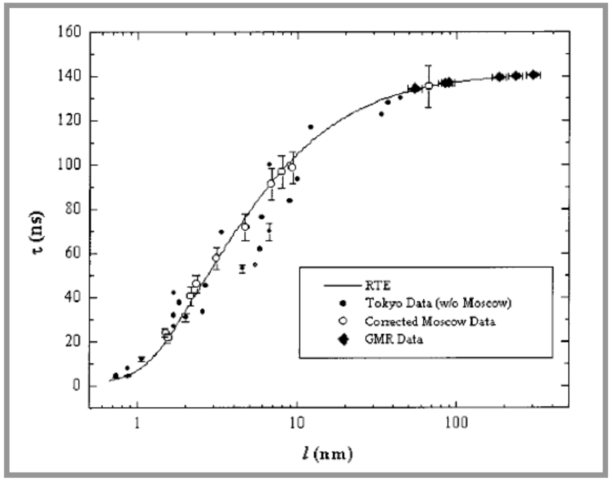
Modules:
J.P. Wrote main and subroutine which initialized N particles in square box.
W.L. Wrote subroutine to calculate net force on a particle, and total potential energy using truncated LJ form.
B.H. Wrote subroutine to propagate positions and velocities using Verlet.
S.L. Created graphical output so that at each step one saw energy, pressure, temperature, and trajectory of particles.
J.L. Wrote thermostat to rescale particle velocities every given number of steps.

Favorite topics

Calculation of physical interest ->
Application to our research

Shooting calculation of $\Psi_n(\mathbf{r})$->
*calculation of the **lifetimes of positrons in solids***

Positron Annhilation Lifetime spectroscopy (PALS) indicates size distribution,contents, and chemical nature of voids



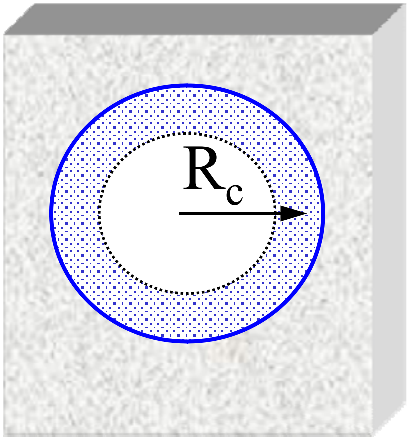
PALS data from silicas and zeolites (Dull, 2001)

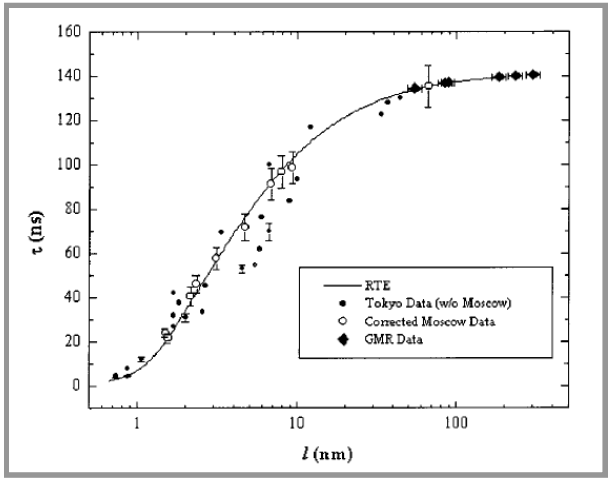
Favorite topics

Calculation of physical interest ->
Application to our research

Shooting calculation of $\Psi_n(\mathbf{r})$->
calculation of the ***lifetimes of positrons in solids***

Brandt-Tao-Eldrup Model:



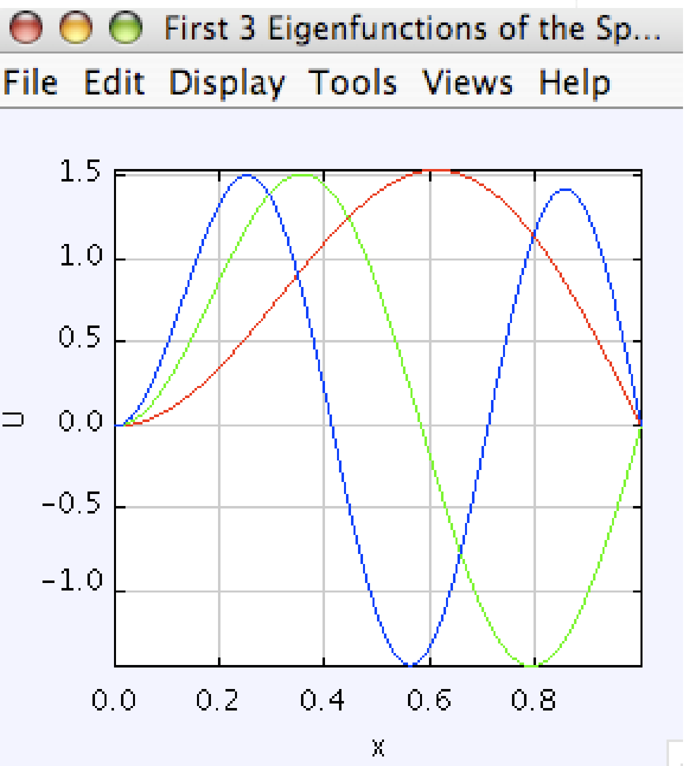


$$\Gamma_{\text{p.o.wall}} = (2ns^{-1}) \int_{r=R_c-\Delta}^{r=R_c} n_+(\mathbf{r})d^3r$$

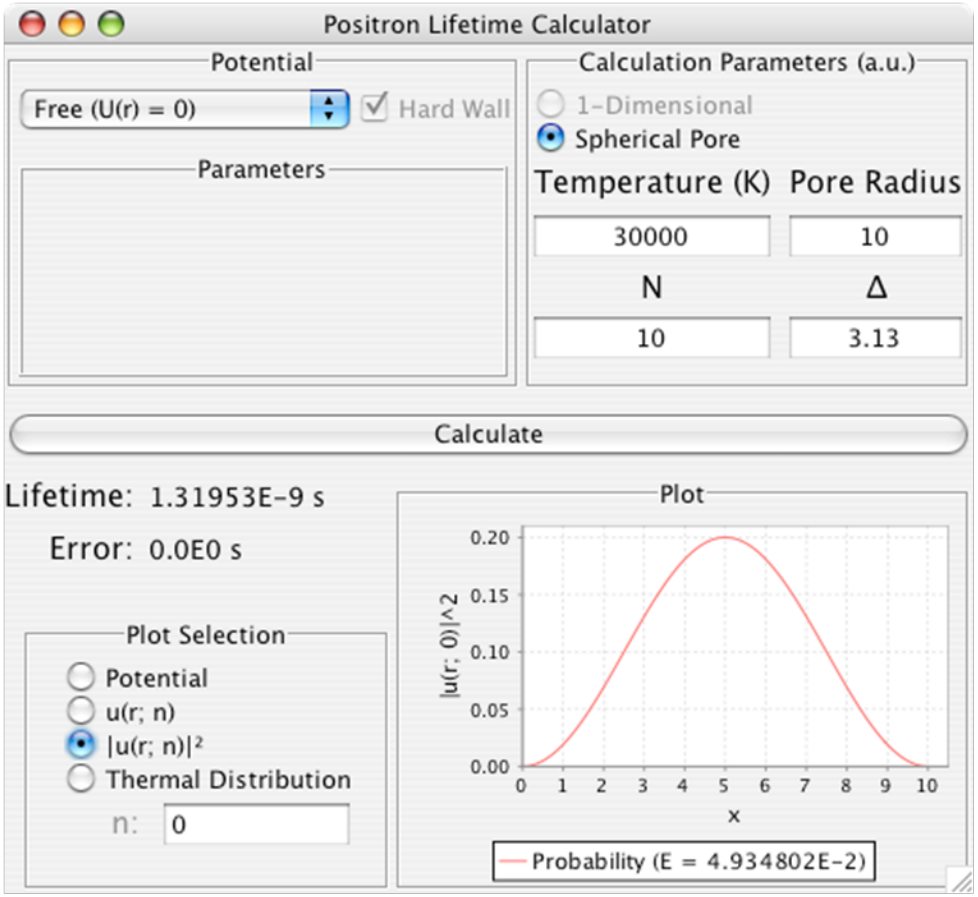
Shooting calculation of $\Psi_n(\mathbf{r})$->
calculation of the lifetimes of positrons in solids

Shooting calculation of $\Psi_n(\mathbf{r})$-> calculation of the lifetimes of positrons in solids

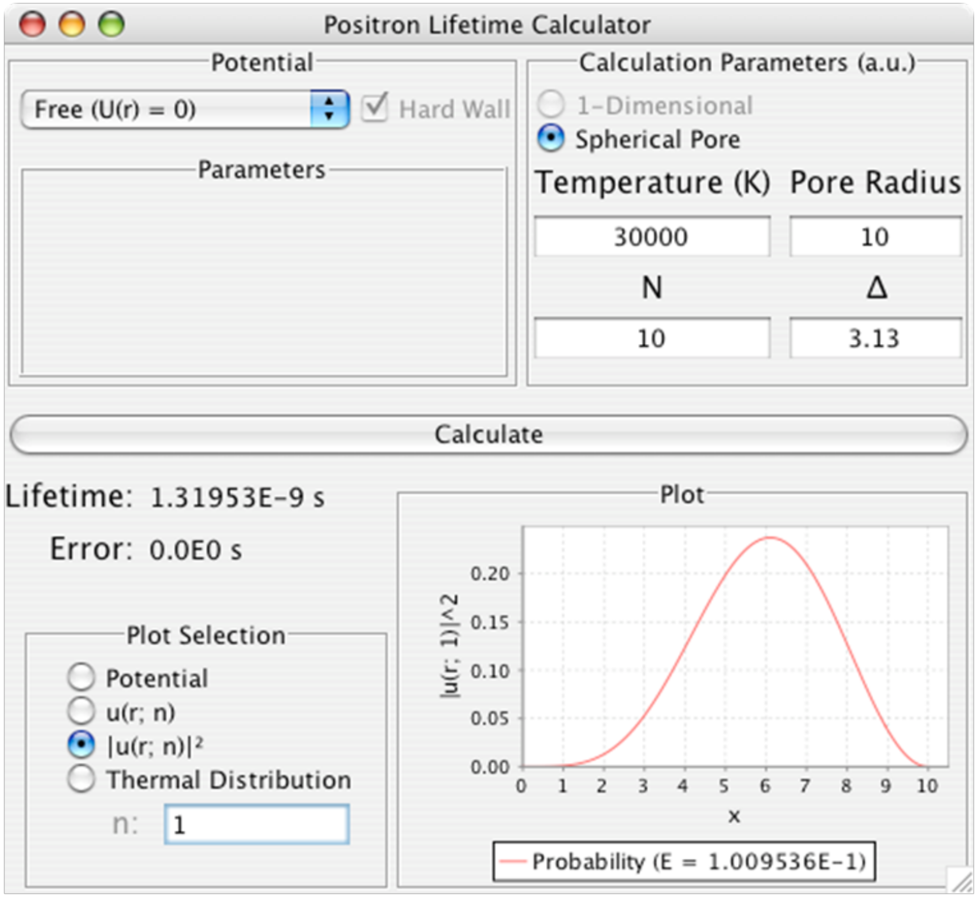
```
package edu.swarthmore.javalifetime.oneD.bvpSolvers;

import org.opensourcephysics.display.Dataset;

public class SphericalBoxTester
{
    public static void main(String[] args)
    {
        int n = 3;
        double radius = 1;
        int l = 1;
        PlotFrame plot = new PlotFrame("x", "$U$", "First " + n
                + " Eigenfunctions of the Spherical Box (radius = " + radius + ", l = "
                + l + ")");
```



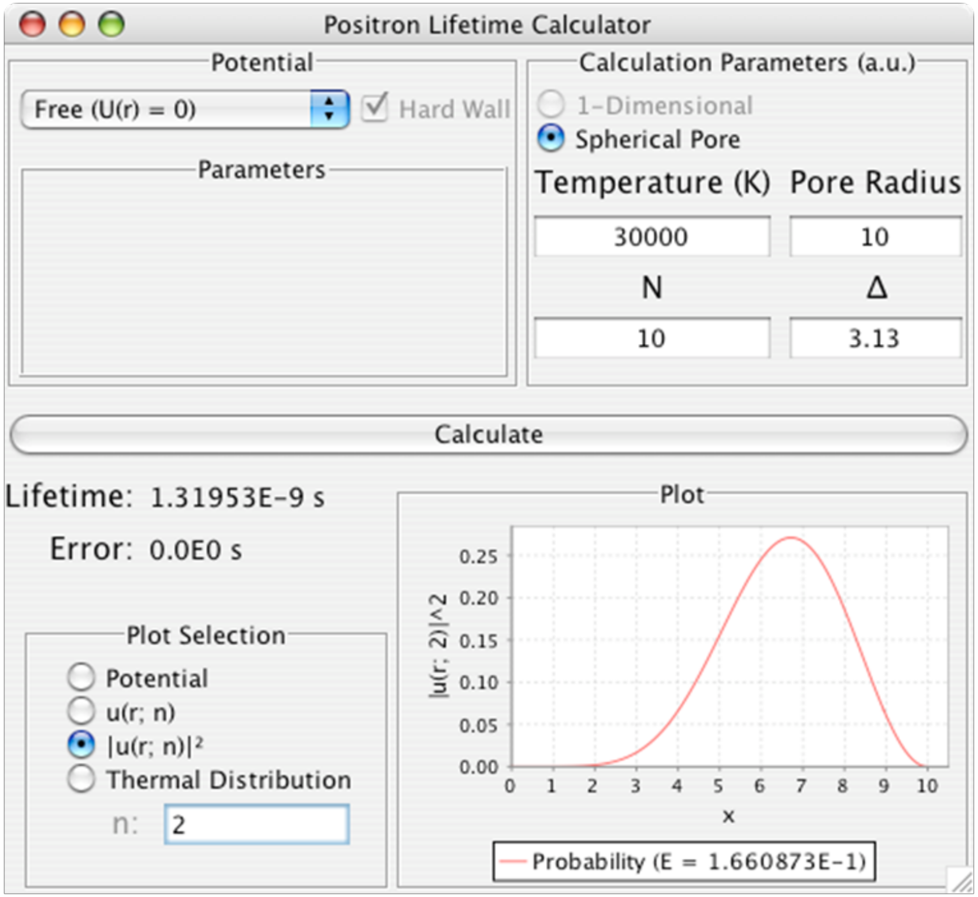
Shooting calculation of $\Psi_n(\mathbf{r})$->
calculation of the lifetimes of positrons in solids



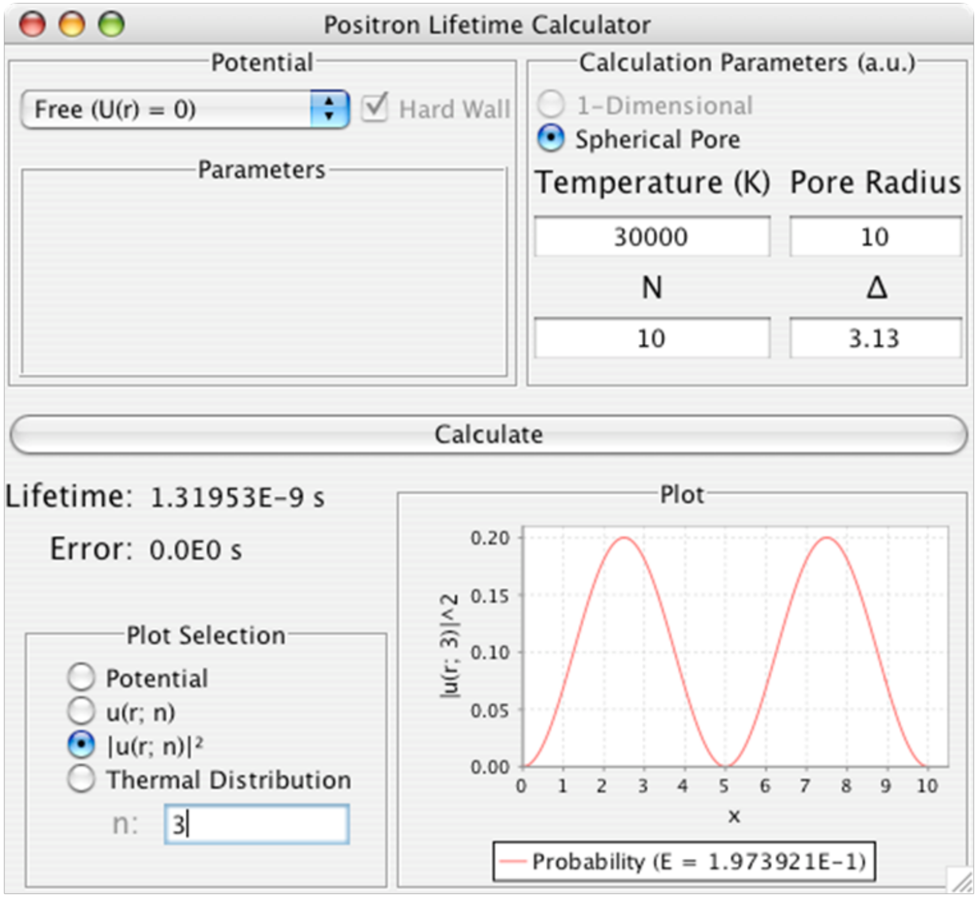
Shooting calculation of $\Psi_n(\mathbf{r})$-> calculation of the lifetimes of positrons in solids



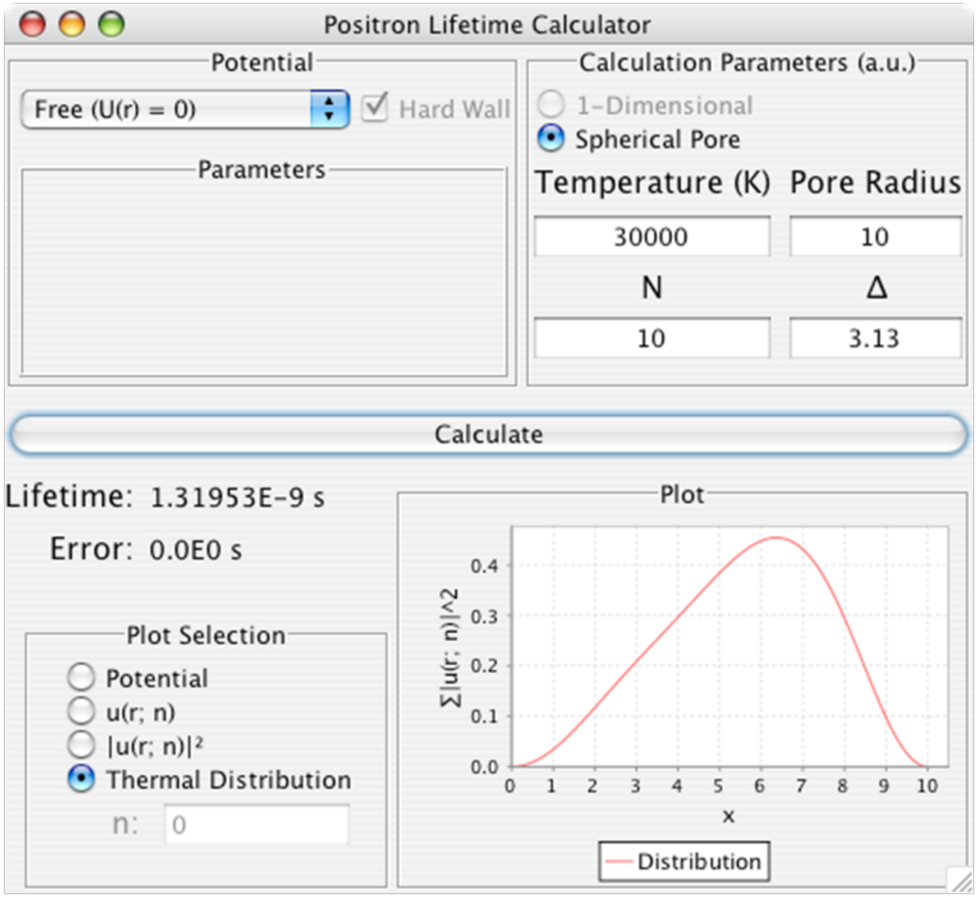
Shooting calculation of $\Psi_n(\mathbf{r})$-> calculation of the lifetimes of positrons in solids



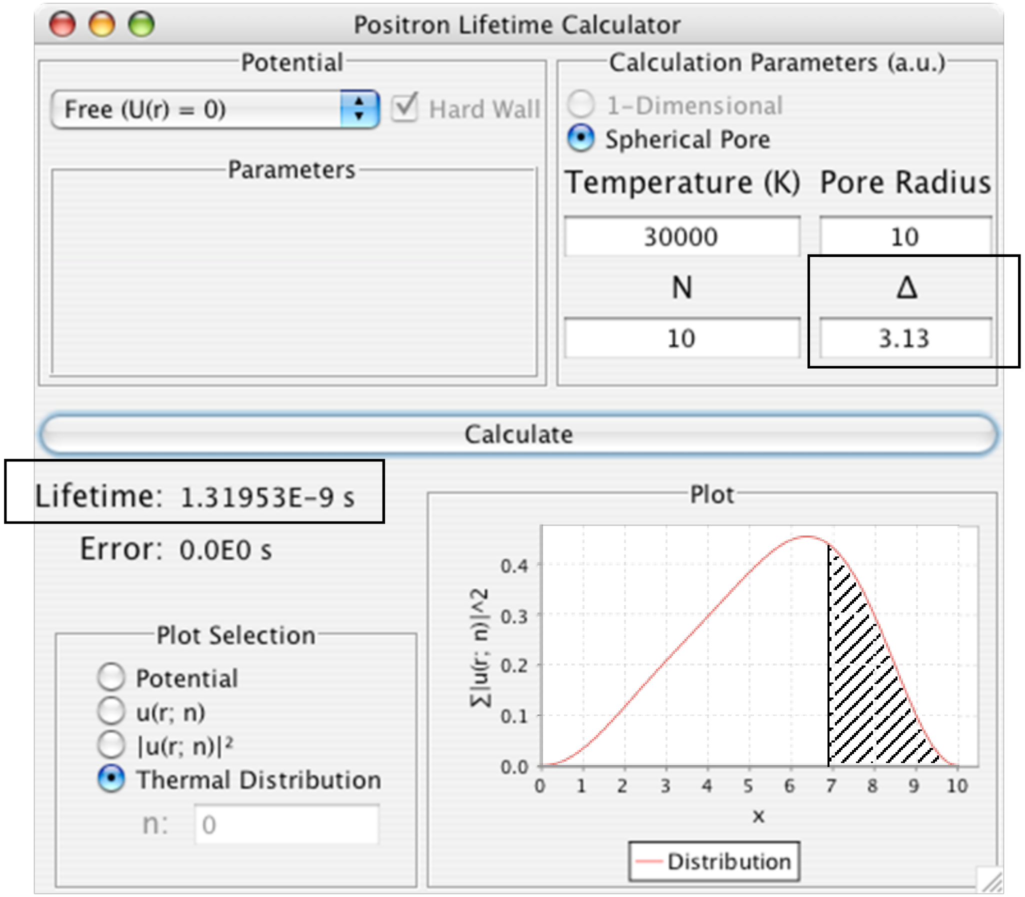
Shooting calculation of $\Psi_n(\mathbf{r})$-> calculation of the lifetimes of positrons in solids



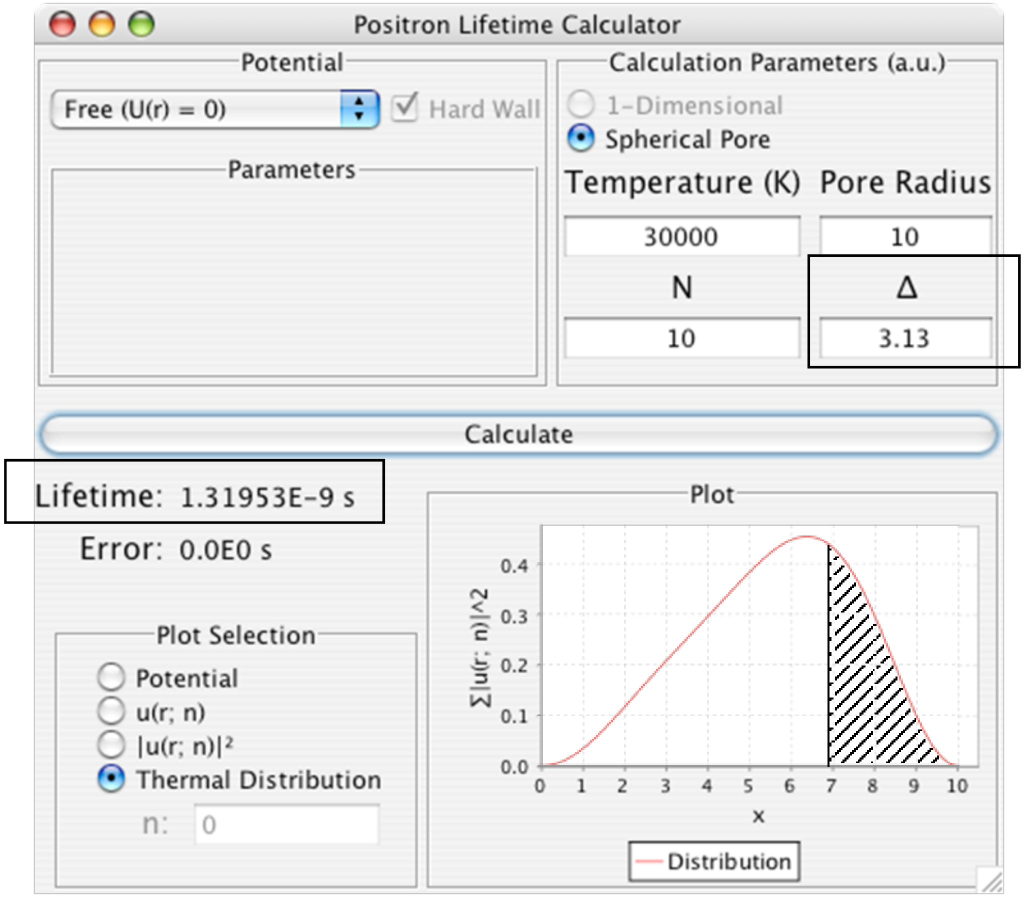
Shooting calculation of $\Psi_n(\mathbf{r})$->
calculation of the lifetimes of positrons in solids



Shooting calculation of $\Psi_n(\mathbf{r})$->
calculation of the lifetimes of positrons in solids



Shooting calculation of $\Psi_n(\mathbf{r})$-> calculation of the lifetimes of positrons in solids



Application assumes spherical pore. It is a mildly useful tool, but is important to us as a stepping stone to a more interesting application, which treats an ellipsoidal pore.

Favorite topics

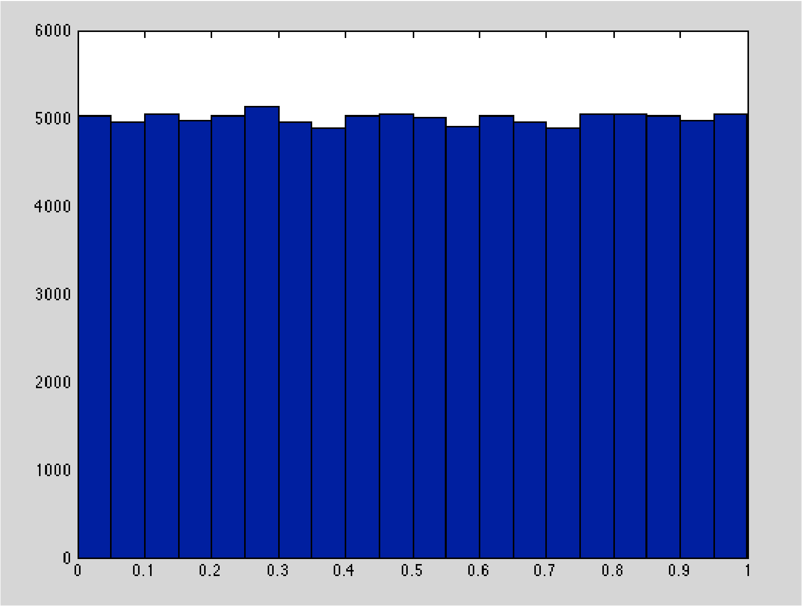
Calculation of physical interest ->
Application to our research

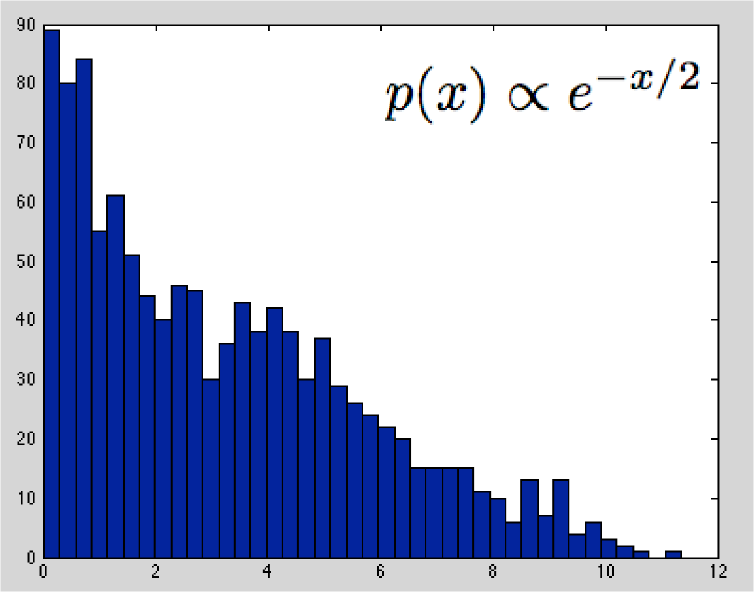
*Monte Carlo calculation of P(**r**; T) ->*
*calculation of the **lifetimes of positrons in solids***

Monte Carlo calculation of P(**r**; T) ->
calculation of the lifetimes of positrons in solids

To sample …

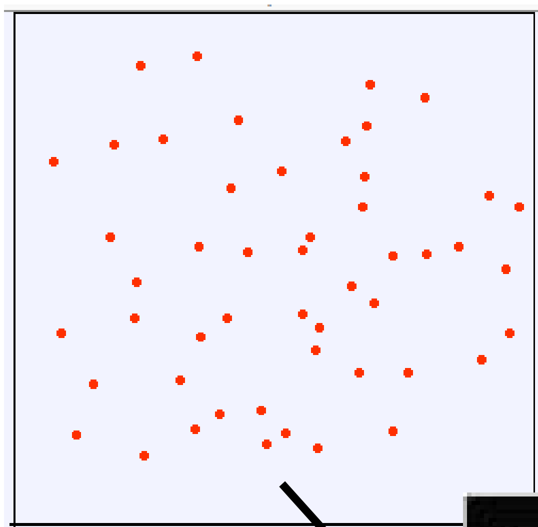
→ *Collection of uniform random numbers.*
Use MC to create nonuniform distribution



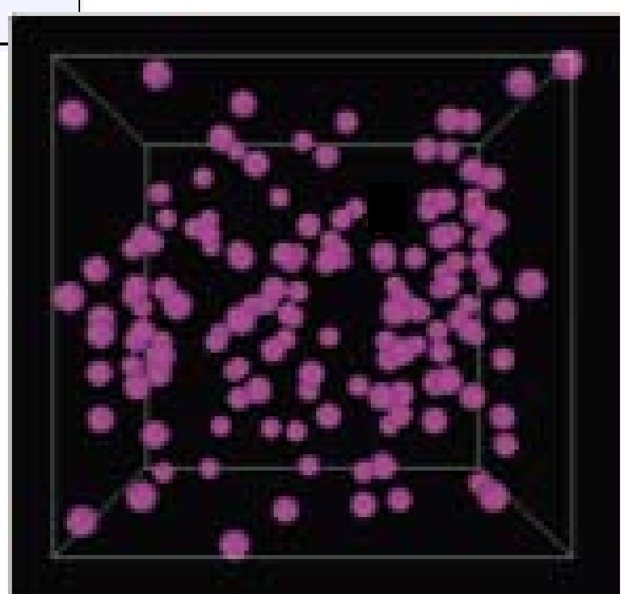


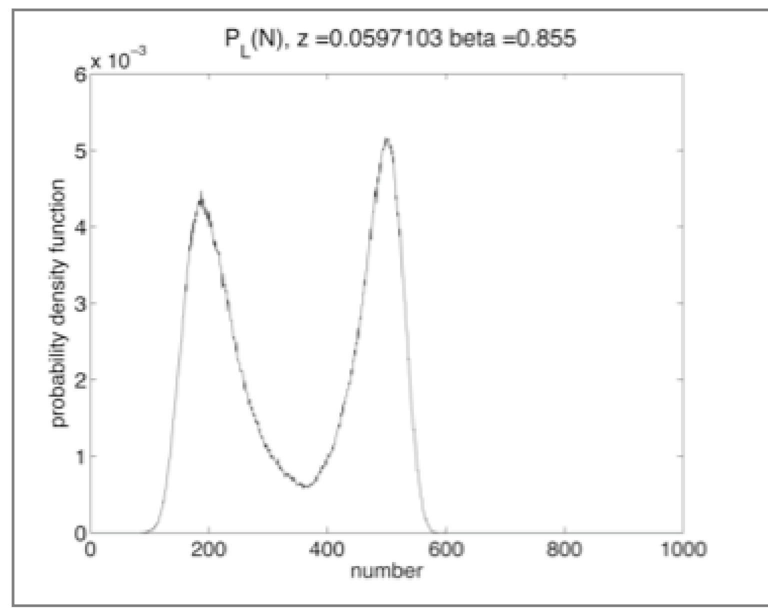
*Monte Carlo calculation of P(**r**; T) ->*
calculation of the lifetimes of positrons in solids

To sample …



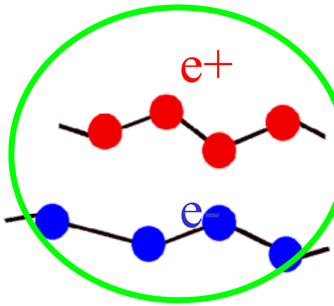
Locations of classical fluid atoms at temperature T.
Use MMC or GCMC to create thermal distribution

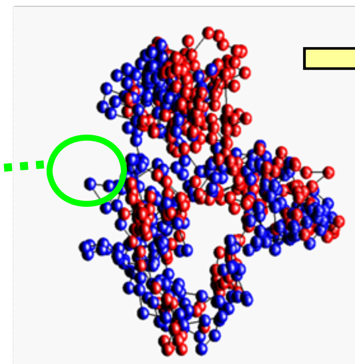




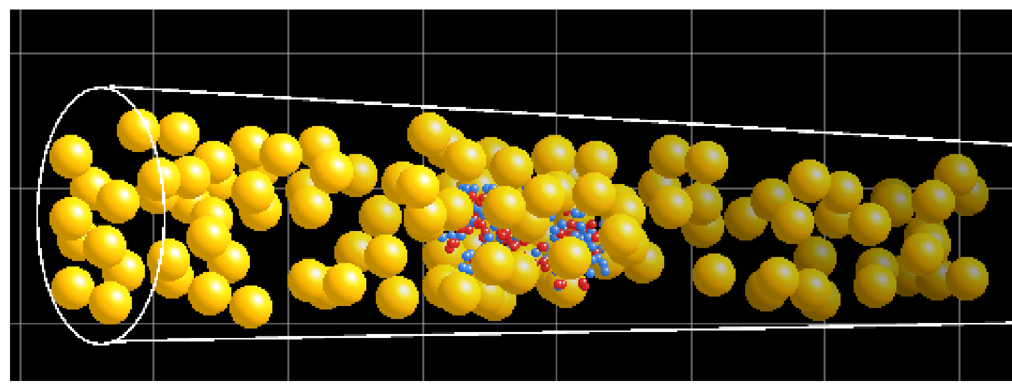
*Monte Carlo calculation of P(**r**; T) ->*
calculation of the lifetimes of positrons in solids

To sample …





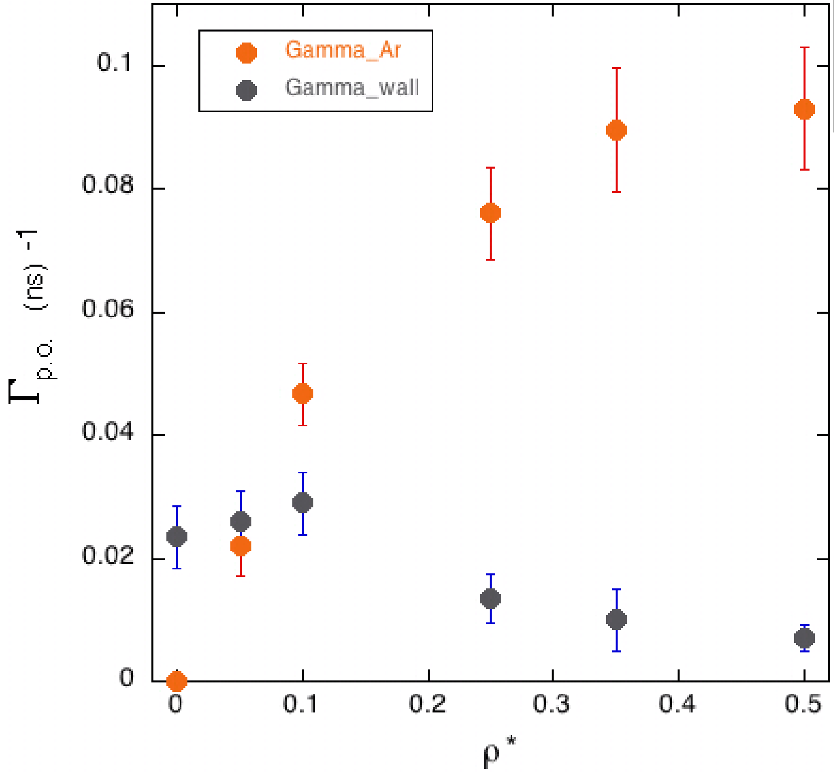
Locations of positron, electron and fluid atoms at temperature T.
Use both PIMC and classical MC to create thermal distribution .

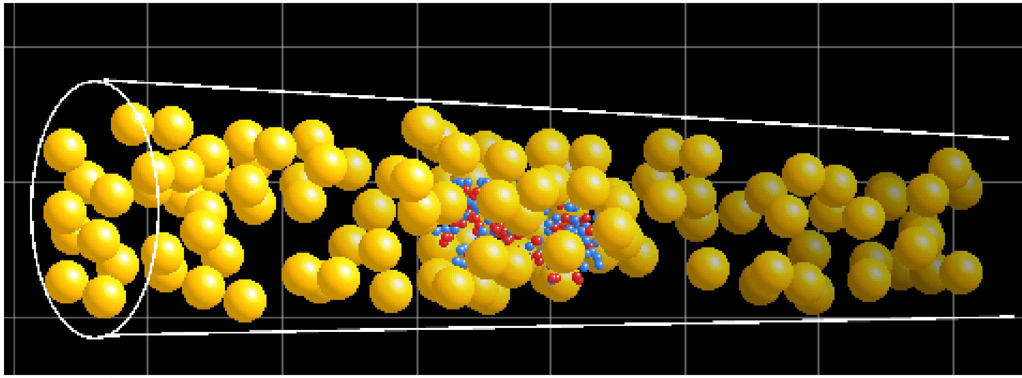


*Monte Carlo calculation of P(**r**; T) ->*
calculation of the lifetimes of positrons in solids

To sample …

Locations of positron, electron and fluid atoms at temperature T.
Use both PIMC and classical MC to create thermal distribution .





$$\Gamma \approx \pi r_e^2 c \int d\mathbf{r}_- \, d\mathbf{r}_+ \, \rho_+(\mathbf{r}_+) \, \rho_-(\mathbf{r}_-) \, \gamma[\rho_-(\mathbf{r}_-)] \, \delta^3(\mathbf{r}_- - \mathbf{r}_+)$$

Many thanks to ...

Colleagues:

Wolfgang Christian (Davidson) and the other organizers of this conference

John Boccio (Swarthmore)

Roy Pollock (LLNL), Terrence Reese (Southern U) , Phil Sterne (LLNL)

Recent Research Students at Swarthmore:

Lisa Larrimore, Robert McFarland, Peter Hastings, Jillian Waldman,

Tim Cronin, Zach Wolfson, Jenny Barry, George Hang, Eric Astor

Funding agencies:

Provost's office of Swarthmore College

PRF

LLNL

AAPT