

Integrating computations, mathematics, physics and chemistry in undergraduate biology programs

Morten Hjorth-Jensen^{1,2} Hans Petter Langtangen^{3,4}

Dept. of Physics, UiO¹

Dept. of Physics, Michigan State University²

Dept. of Informatics, UiO³

Center for Biomedical Computing, Simula⁴

Oct 18, 2014

Overarching questions

- Which skills are needed by MSc candidates in biology?
- There is new demand for more
 - quantitative methods & reasoning
 - computational approaches and programming
 - data analysis and visualization
- How to integrate such computing-based activities in the bachelor programs?

Overarching questions

- Which skills are needed by MSc candidates in biology?
- There is new demand for more
 - quantitative methods & reasoning
 - understanding data and phenomena via models
- How to integrate such computing-based activities in the bachelor programs?

Overarching questions

- Which skills are needed by MSc candidates in biology?
- There is new demand for more
 - quantitative methods & reasoning
 - understanding data and phenomena via models
 - creating *in silico* virtual labs
- How to integrate such computing-based activities in the bachelor programs?

Overarching questions

- Which skills are needed by MSc candidates in biology?
- There is new demand for more
 - quantitative methods & reasoning
 - understanding data and phenomena via models
 - creating *in silico* virtual labs
- How to integrate such computing-based activities in the bachelor programs?

Overarching questions

- Which skills are needed by MSc candidates in biology?
- There is new demand for more
 - quantitative methods & reasoning
 - understanding data and phenomena via models
 - creating *in silico* virtual labs
- How to integrate such computing-based activities in the bachelor programs?

Overarching questions

- Which skills are needed by MSc candidates in biology?
- There is new demand for more
 - quantitative methods & reasoning
 - understanding data and phenomena via models
 - creating *in silico* virtual labs
- How to integrate such computing-based activities in the bachelor programs?

Overarching questions

- Which skills are needed by MSc candidates in biology?
- There is new demand for more
 - quantitative methods & reasoning
 - understanding data and phenomena via models
 - creating *in silico* virtual labs
- How to integrate such computing-based activities in the bachelor programs?

How to teach computing in biology?

- Do we need to still following the tradition and teach mathematics, physics, computations, chemistry, etc. in separate discipline-specific courses?
- Results:
 - uninteresting to first study tools when you want to study biology
 - difficult to learn tools when you are not motivated
 - tools still seen as tools, not as biology
- It's time for new thinking:
 - *Computational Biology* is not just a new course, but a new way of thinking
 - *Computational Biology* is not just a new course, but a new way of thinking
 - *Computational Biology* is not just a new course, but a new way of thinking
- Corollary: this new thinking can be applied in chemistry, geology, and all non-math-oriented programs

How to teach computing in biology?

- Do we need to still following the tradition and teach mathematics, physics, computations, chemistry, etc. in separate discipline-specific courses?
- Results:
 - uninteresting to first study tools when you want to study biology
 - little understanding of what the tools are good for
- It's time for new thinking:
- Corollary: this new thinking can be applied in chemistry, geology, ... and all non-math-oriented programs

How to teach computing in biology?

- Do we need to still following the tradition and teach mathematics, physics, computations, chemistry, etc. in separate discipline-specific courses?
- Results:
 - uninteresting to first study tools when you want to study biology
 - little understanding of what the tools are good for
 - minor utilization of tools later in biology
- It's time for new thinking:
- Corollary: this new thinking can be applied in chemistry, geology, ... and all non-math-oriented programs

How to teach computing in biology?

- Do we need to still following the tradition and teach mathematics, physics, computations, chemistry, etc. in separate discipline-specific courses?
- Results:
 - uninteresting to first study tools when you want to study biology
 - little understanding of what the tools are good for
 - minor utilization of tools later in biology
- It's time for new thinking:
- Corollary: this new thinking can be applied in chemistry, geology, ... and all non-math-oriented programs

How to teach computing in biology?

- Do we need to still following the tradition and teach mathematics, physics, computations, chemistry, etc. in separate discipline-specific courses?
- Results:
 - uninteresting to first study tools when you want to study biology
 - little understanding of what the tools are good for
 - minor utilization of tools later in biology
- It's time for new thinking:

→ how to teach computing, math tools, etc. in biology

- Corollary: this new thinking can be applied in chemistry, geology, ... and all non-math-oriented programs

How to teach computing in biology?

- Do we need to still following the tradition and teach mathematics, physics, computations, chemistry, etc. in separate discipline-specific courses?
- Results:
 - uninteresting to first study tools when you want to study biology
 - little understanding of what the tools are good for
 - minor utilization of tools later in biology
- It's time for new thinking:
 - Just-in-time teaching: teach tools *when needed*
 - Teach tools in the context of biology
- Corollary: this new thinking can be applied in chemistry, geology, ... and all non-math-oriented programs

How to teach computing in biology?

- Do we need to still following the tradition and teach mathematics, physics, computations, chemistry, etc. in separate discipline-specific courses?
- Results:
 - uninteresting to first study tools when you want to study biology
 - little understanding of what the tools are good for
 - minor utilization of tools later in biology
- It's time for new thinking:
 - Just-in-time teaching: teach tools *when needed*
 - Teach tools in the *context of biology*
 - Emphasize development of *intuition and understanding*
 - Base learning of the students' own *explorations*
 - Base teaching on examples and projects (from biology)
 - Integrate lab work with computing tools
 - Teach tools in BIOxxxx courses - this emphasizes the importance of computing, physics, mathematics, ... in biology
- Corollary: this new thinking can be applied in chemistry, geology, ... and all non-math-oriented programs

How to teach computing in biology?

- Do we need to still following the tradition and teach mathematics, physics, computations, chemistry, etc. in separate discipline-specific courses?
- Results:
 - uninteresting to first study tools when you want to study biology
 - little understanding of what the tools are good for
 - minor utilization of tools later in biology
- It's time for new thinking:
 - Just-in-time teaching: teach tools *when needed*
 - Teach tools in the *context of biology*
 - Emphasize development of *intuition and understanding*
 - Base learning of the students' own *explorations*
 - Base teaching on examples and projects (from biology)
 - Integrate lab work with computing tools
 - Teach tools in BIOxxxx courses - this emphasizes the importance of computing, physics, mathematics, ... in biology
- Corollary: this new thinking can be applied in chemistry, geology, ... and all non-math-oriented programs

How to teach computing in biology?

- Do we need to still following the tradition and teach mathematics, physics, computations, chemistry, etc. in separate discipline-specific courses?
- Results:
 - uninteresting to first study tools when you want to study biology
 - little understanding of what the tools are good for
 - minor utilization of tools later in biology
- It's time for new thinking:
 - Just-in-time teaching: teach tools *when needed*
 - Teach tools in the *context of biology*
 - Emphasize development of *intuition and understanding*
 - Base learning of the students' own *explorations*
 - Base teaching on examples and projects (from biology)
 - Integrate lab work with computing tools
 - Teach tools in BIOxxxx courses - this emphasizes the importance of computing, physics, mathematics, ... in biology
- Corollary: this new thinking can be applied in chemistry, geology, ... and all non-math-oriented programs

How to teach computing in biology?

- Do we need to still following the tradition and teach mathematics, physics, computations, chemistry, etc. in separate discipline-specific courses?
- Results:
 - uninteresting to first study tools when you want to study biology
 - little understanding of what the tools are good for
 - minor utilization of tools later in biology
- It's time for new thinking:
 - Just-in-time teaching: teach tools *when needed*
 - Teach tools in the *context of biology*
 - Emphasize development of *intuition and understanding*
 - Base learning of the students' own *explorations*
 - Base teaching on examples and projects (from biology)
 - Integrate lab work with computing tools
 - Teach tools in BIOxxxx courses - this emphasizes the importance of computing, physics, mathematics, ... in biology
- Corollary: this new thinking can be applied in chemistry, biology, ... and all non-math-oriented programs

How to teach computing in biology?

- Do we need to still following the tradition and teach mathematics, physics, computations, chemistry, etc. in separate discipline-specific courses?
- Results:
 - uninteresting to first study tools when you want to study biology
 - little understanding of what the tools are good for
 - minor utilization of tools later in biology
- It's time for new thinking:
 - Just-in-time teaching: teach tools *when needed*
 - Teach tools in the *context of biology*
 - Emphasize development of *intuition and understanding*
 - Base learning of the students' own *explorations*
 - Base teaching on examples and projects (from biology)
 - Integrate lab work with computing tools
 - Teach tools in BIOxxxx courses - this emphasizes the importance of computing, physics, mathematics, ... in biology
- Corollary: this new thinking can be applied in chemistry, biology, ... and all non-math-oriented programs

How to teach computing in biology?

- Do we need to still following the tradition and teach mathematics, physics, computations, chemistry, etc. in separate discipline-specific courses?
- Results:
 - uninteresting to first study tools when you want to study biology
 - little understanding of what the tools are good for
 - minor utilization of tools later in biology
- It's time for new thinking:
 - Just-in-time teaching: teach tools *when needed*
 - Teach tools in the *context of biology*
 - Emphasize development of *intuition and understanding*
 - Base learning of the students' own *explorations*
 - Base teaching on examples and projects (from biology)
 - Integrate lab work with computing tools
 - Teach tools in BIOxxxx courses - this emphasizes the importance of computing, physics, mathematics, ... in biology
- Corollary: this new thinking can be applied in chemistry, geology ... and all non-math-oriented programs

How to teach computing in biology?

- Do we need to still following the tradition and teach mathematics, physics, computations, chemistry, etc. in separate discipline-specific courses?
- Results:
 - uninteresting to first study tools when you want to study biology
 - little understanding of what the tools are good for
 - minor utilization of tools later in biology
- It's time for new thinking:
 - Just-in-time teaching: teach tools *when needed*
 - Teach tools in the *context of biology*
 - Emphasize development of *intuition and understanding*
 - Base learning of the students' own *explorations*
 - Base teaching on examples and projects (from biology)
 - Integrate lab work with computing tools
 - Teach tools in BIOxxxx courses - this emphasizes the importance of computing, physics, mathematics, ... in biology
- Corollary: this new thinking can be applied in chemistry, geology ... and all non-math-oriented programs

How to teach computing in biology?

- Do we need to still following the tradition and teach mathematics, physics, computations, chemistry, etc. in separate discipline-specific courses?
- Results:
 - uninteresting to first study tools when you want to study biology
 - little understanding of what the tools are good for
 - minor utilization of tools later in biology
- It's time for new thinking:
 - Just-in-time teaching: teach tools *when needed*
 - Teach tools in the *context of biology*
 - Emphasize development of *intuition and understanding*
 - Base learning of the students' own *explorations*
 - Base teaching on examples and projects (from biology)
 - Integrate lab work with computing tools
 - Teach tools in BIOxxxx courses - this emphasizes the importance of computing, physics, mathematics, ... in biology
- Corollary: this new thinking can be applied in chemistry, geology ... and all non-math-oriented programs

How to teach computing in biology?

- Do we need to still following the tradition and teach mathematics, physics, computations, chemistry, etc. in separate discipline-specific courses?
- Results:
 - uninteresting to first study tools when you want to study biology
 - little understanding of what the tools are good for
 - minor utilization of tools later in biology
- It's time for new thinking:
 - Just-in-time teaching: teach tools *when needed*
 - Teach tools in the *context of biology*
 - Emphasize development of *intuition and understanding*
 - Base learning of the students' own *explorations*
 - Base teaching on examples and projects (from biology)
 - Integrate lab work with computing tools
 - Teach tools in BIOxxxx courses - this emphasizes the importance of computing, physics, mathematics, ... in biology
- Corollary: this new thinking can be applied in chemistry, geology and all non-math-oriented programs

Current status

- CSE (*Computing in Science Education*) in life science is a faculty priority area
- Core professors in the CSE project and 5 IBV professors are dedicated to implementation
- 4 CINPLA PhD students have excellent background for and strong interests in the implementation
- Let's roll!

Possible pilot project

- 1 year: Oct 2014 - Oct 2015
- Form a project team of dedicated CSE+IBV people and CINPLA PhD students
- Develop a pedagogical framework
- CSE+IBV people identify a set of possible examples
- PhD students work closely with CSE+IBV people to find data, define models, and write documents
- Intro course in computing and programming for IBV teachers, using selected examples (spring 2015)
- Educational workshop at IBV:
 - Present results from the project
 - Discuss how and where to implement examples in BIOxxxx courses
- Fall 2016: First integration of computing in courses

The pedagogical framework

Aim: Develop student intuition about the scientific method

- Method: problem-based learning
- Coherent problem solving *in biology* by integrating mathematics, programming, physics/chemistry, ...
- Starting point: data from lab or field experiments
- Visualize data
- Derive computational models directly from mathematical reasoning
- Program models, fit parameters, compare with data
- Develop intuition and understanding based on
 - the principles behind the model
 - exploration of the model ("what if")
 - prediction of new experiments
- For each example, document this scientific method in a common, structured framework

Example 1: ecoli lab experiment

Observations of no of bacteria vs time in seconds, stored in Excel and written to a CVS file:

```
0,100  
600,140  
1200,250  
1800,360  
2400,480  
3000,820  
3600,1300  
4200,1700  
4800,2900  
5400,3900  
6000,7000
```

Visualize data

- Meet a text editor and a terminal window
- Very basic Unix

First program:

```
t = [0, 600, 1200, 1800, 2400, 3000, 3600,  
     4200, 4800, 5400, 6000]  
N = [100, 140, 250, 360, 480, 820, 1300, 1700, 2900, 3900, 7000]  
import matplotlib.pyplot as plt  
plt.plot(t, N, 'ro')  
plt.xlabel('t [s]')  
plt.ylabel('N')  
plt.show()
```

Concepts that are implicitly introduced

- Lists or arrays of numbers
- Plotting commands
- Curve
- Function of time

The concept of a continuous function $N(t)$ is not necessary. All we need for a long time is the concept of a curve: straight lines between points.

Read data from file

```
import numpy as np
data = np.loadtxt('ecoli.dat', delimiter=',')
print data
t = data[:,0]
N = data[:,1]
import matplotlib.pyplot as plt
plt.plot(t, N, 'ro')
plt.xlabel('t [s]')
plt.ylabel('N')
plt.show()
```

Typical pattern:

The population grows faster and faster.

Use IPython notebook as lab journal.

How can we reason about the process?

- 1 Cells divide after T seconds on average (one generation)
- 2 $2N$ cells divide on twice as many new cells ΔN in a time interval Δt as N cells would: $\Delta N \propto N$
- 3 N cells result in twice as many new individuals ΔN in time $2\Delta t$ as in time Δt : $\Delta N \propto \Delta t$
- 4 Same proportionality wrt death (repeat reasoning)
- 5 Proposed model: $\Delta N = b\Delta tN - d\Delta tN$ for some unknown constants b (births) and d (deaths)
- 6 Describe evolution in discrete time: $t_n = n\Delta t$
- 7 Program-friendly notation: N at t_n is N^n
- 8 Math model: $N^{n+1} = N^n + r\Delta N$ ($r = b - d$)
- 9 Program model: $N[n+1] = N[n] + r*dt*N[n]$

How can we reason about the process?

- 1 Cells divide after T seconds on average (one generation)
- 2 $2N$ cells divide otn twice as many new cells ΔN in a time interval Δt as N cells would: $\Delta N \propto N$
- 3 N cells result in twice as many new individuals ΔN in time $2\Delta t$ as in time Δt : $\Delta N \propto \Delta t$
- 4 Same proportionality wrt death (repeat reasoning)
- 5 Proposed model: $\Delta N = b\Delta tN - d\Delta tN$ for some unknown constants b (births) and d (deaths)
- 6 Describe evolution in discrete time: $t_n = n\Delta t$
- 7 Program-friendly notation: N at t_n is N^n
- 8 Math model: $N^{n+1} = N^n + r\Delta N$ ($r = b - d$)
- 9 Program model: $N[n+1] = N[n] + r*dt*N[n]$

How can we reason about the process?

- 1 Cells divide after T seconds on average (one generation)
- 2 $2N$ cells divide on twice as many new cells ΔN in a time interval Δt as N cells would: $\Delta N \propto N$
- 3 N cells result in twice as many new individuals ΔN in time $2\Delta t$ as in time Δt : $\Delta N \propto \Delta t$
- 4 Same proportionality wrt death (repeat reasoning)
- 5 Proposed model: $\Delta N = b\Delta tN - d\Delta tN$ for some unknown constants b (births) and d (deaths)
- 6 Describe evolution in discrete time: $t_n = n\Delta t$
- 7 Program-friendly notation: N at t_n is N^n
- 8 Math model: $N^{n+1} = N^n + r\Delta N$ ($r = b - d$)
- 9 Program model: $N[n+1] = N[n] + r*dt*N[n]$

How can we reason about the process?

- 1 Cells divide after T seconds on average (one generation)
- 2 $2N$ cells divide on twice as many new cells ΔN in a time interval Δt as N cells would: $\Delta N \propto N$
- 3 N cells result in twice as many new individuals ΔN in time $2\Delta t$ as in time Δt : $\Delta N \propto \Delta t$
- 4 Same proportionality wrt death (repeat reasoning)
- 5 Proposed model: $\Delta N = b\Delta tN - d\Delta tN$ for some unknown constants b (births) and d (deaths)
- 6 Describe evolution in discrete time: $t_n = n\Delta t$
- 7 Program-friendly notation: N at t_n is N^n
- 8 Math model: $N^{n+1} = N^n + r\Delta tN^n$ ($r = b - d$)
- 9 Program model: $N[n+1] = N[n] + r*dt*N[n]$

How can we reason about the process?

- 1 Cells divide after T seconds on average (one generation)
- 2 $2N$ cells divide on twice as many new cells ΔN in a time interval Δt as N cells would: $\Delta N \propto N$
- 3 N cells result in twice as many new individuals ΔN in time $2\Delta t$ as in time Δt : $\Delta N \propto \Delta t$
- 4 Same proportionality wrt death (repeat reasoning)
- 5 Proposed model: $\Delta N = b\Delta tN - d\Delta tN$ for some unknown constants b (births) and d (deaths)
- 6 Describe evolution in discrete time: $t_n = n\Delta t$
- 7 Program-friendly notation: N at t_n is N^n
- 8 Math model: $N^{n+1} = N^n + r\Delta N$ ($r = b - d$)
- 9 Program model: $N[n+1] = N[n] + r*dt*N[n]$

How can we reason about the process?

- 1 Cells divide after T seconds on average (one generation)
- 2 $2N$ cells divide on twice as many new cells ΔN in a time interval Δt as N cells would: $\Delta N \propto N$
- 3 N cells result in twice as many new individuals ΔN in time $2\Delta t$ as in time Δt : $\Delta N \propto \Delta t$
- 4 Same proportionality wrt death (repeat reasoning)
- 5 Proposed model: $\Delta N = b\Delta tN - d\Delta tN$ for some unknown constants b (births) and d (deaths)
- 6 Describe evolution in discrete time: $t_n = n\Delta t$
- 7 Program-friendly notation: N at t_n is N^n
- 8 Math model: $N^{n+1} = N^n + r\Delta N$ ($r = b - d$)
- 9 Program model: $N[n+1] = N[n] + r*dt*N[n]$

How can we reason about the process?

- 1 Cells divide after T seconds on average (one generation)
- 2 $2N$ cells divide on twice as many new cells ΔN in a time interval Δt as N cells would: $\Delta N \propto N$
- 3 N cells result in twice as many new individuals ΔN in time $2\Delta t$ as in time Δt : $\Delta N \propto \Delta t$
- 4 Same proportionality wrt death (repeat reasoning)
- 5 Proposed model: $\Delta N = b\Delta tN - d\Delta tN$ for some unknown constants b (births) and d (deaths)
- 6 Describe evolution in discrete time: $t_n = n\Delta t$
- 7 Program-friendly notation: N at t_n is N^n
- 8 Math model: $N^{n+1} = N^n + r\Delta N$ ($r = b - d$)
- 9 Program model: $N[n+1] = N[n] + r*dt*N[n]$

How can we reason about the process?

- 1 Cells divide after T seconds on average (one generation)
- 2 $2N$ cells divide on twice as many new cells ΔN in a time interval Δt as N cells would: $\Delta N \propto N$
- 3 N cells result in twice as many new individuals ΔN in time $2\Delta t$ as in time Δt : $\Delta N \propto \Delta t$
- 4 Same proportionality wrt death (repeat reasoning)
- 5 Proposed model: $\Delta N = b\Delta tN - d\Delta tN$ for some unknown constants b (births) and d (deaths)
- 6 Describe evolution in discrete time: $t_n = n\Delta t$
- 7 Program-friendly notation: N at t_n is N^n
- 8 Math model: $N^{n+1} = N^n + r\Delta N$ ($r = b - d$)
- 9 Program model: $N[n+1] = N[n] + r*dt*N[n]$

How can we reason about the process?

- 1 Cells divide after T seconds on average (one generation)
- 2 $2N$ cells divide on twice as many new cells ΔN in a time interval Δt as N cells would: $\Delta N \propto N$
- 3 N cells result in twice as many new individuals ΔN in time $2\Delta t$ as in time Δt : $\Delta N \propto \Delta t$
- 4 Same proportionality wrt death (repeat reasoning)
- 5 Proposed model: $\Delta N = b\Delta tN - d\Delta tN$ for some unknown constants b (births) and d (deaths)
- 6 Describe evolution in discrete time: $t_n = n\Delta t$
- 7 Program-friendly notation: N at t_n is N^n
- 8 Math model: $N^{n+1} = N^n + r\Delta N$ ($r = b - d$)
- 9 Program model: $N[n+1] = N[n] + r*dt*N[n]$

How can we reason about the process?

- 1 Cells divide after T seconds on average (one generation)
- 2 $2N$ cells divide on twice as many new cells ΔN in a time interval Δt as N cells would: $\Delta N \propto N$
- 3 N cells result in twice as many new individuals ΔN in time $2\Delta t$ as in time Δt : $\Delta N \propto \Delta t$
- 4 Same proportionality wrt death (repeat reasoning)
- 5 Proposed model: $\Delta N = b\Delta tN - d\Delta tN$ for some unknown constants b (births) and d (deaths)
- 6 Describe evolution in discrete time: $t_n = n\Delta t$
- 7 Program-friendly notation: N at t_n is N^n
- 8 Math model: $N^{n+1} = N^n + r\Delta N$ ($r = b - d$)
- 9 Program model: $N[n+1] = N[n] + r*dt*N[n]$

The first simple program

Let us solve the difference equation in as simple way as possible, just to train some programming: $r = 1.5$, $N^0 = 1$, $\Delta t = 0.5$

```
import numpy as np

t = np.linspace(0, 10, 21) # 20 intervals in [0, 10]
dt = t[1] - t[0]
N = np.zeros(t.size)

N[0] = 1
r = 0.5

for n in range(0, N.size-1, 1):
    N[n+1] = N[n] + r*dt*N[n]
    print 'N[%d]=%.1f' % (n+1, N[n+1])
```

The output

```
N[1]=1.2  
N[2]=1.6  
N[3]=2.0  
N[4]=2.4  
N[5]=3.1  
N[6]=3.8  
N[7]=4.8  
N[8]=6.0  
N[9]=7.5  
N[10]=9.3  
N[11]=11.6  
N[12]=14.6  
N[13]=18.2  
N[14]=22.7  
N[15]=28.4  
N[16]=35.5  
N[17]=44.4  
N[18]=55.5  
N[19]=69.4  
N[20]=86.7
```

Parameter estimation

- We do not know r
- How can we estimate r from data?

We can use the difference equation with the experimental data

$$N^{n+1} = N^n + r\Delta t N^n$$

Say N^{n+1} and N^n are known from data, solve wrt r :

$$r = \frac{N^{n+1} - N^n}{N^n \Delta t}$$

Use experimental data in the fraction, say $t_1 = 600$, $t_2 = 1200$, $N^1 = 140$, $N^2 = 250$: $r = 0.0013$.

More sophisticated methods

Can do a nonlinear least squares parameter fit, but that is too advanced at this stage.

A program relevant for the biological problem

```
import numpy as np

# Estimate r
data = np.loadtxt('ecoli.csv', delimiter=',')
t_e = data[:,0]
N_e = data[:,1]
i = 3
r = (N_e[i+1] - N_e[i]) / (N_e[i] * (t_e[i+1] - t_e[i]))
# Can experiment with r values and see if the model can
# match the data

T = 1200      # cell can divide after T sec
t_max = 5*T   # 5 generations in experiment
t = np.linspace(0, t_max, 1000)
dt = t[1] - t[0]
N = np.zeros(t.size)

N[0] = 100
for n in range(0, len(t)-1, 1):
    N[n+1] = N[n] + r*dt*N[n]
    print N[n+1]

import matplotlib.pyplot as plt
plt.plot(t, N, 'r-', t_e, N_e, 'bo')
plt.xlabel('time [s]'); plt.ylabel('N')
plt.legend(['model', 'experiment'])
plt.show()
```

Discuss the nature of such a model

- Write up all the biological factors that influence the population size of bacteria
- Understand that all such effects are merged into one parameter r
- Understand that the reasoning must be the same whether we have bacteria, animals or humans - this is a generic model!

Adding model complexity: Predator-Prey model from ecology

The population dynamics of a simple predator-prey system is a classical example shown in many biology textbooks when ecological systems are discussed. The system contains all elements of the scientific method:

- The set up of a specific hypothesis combined with
- the experimental methods needed (one can study existing data or perform experiments)
- analyzing and interpreting the data and performing further experiments if needed
- trying to extract general behaviors and extract eventual laws or patterns
- develop mathematical relations for the uncovered regularities/laws and test these by performing new experiments

Case study from Hudson bay

Lots of data about populations of hares and lynx collected from furs in Hudson Bay, Canada, are available. It is known that the populations oscillate. Why? We shall demonstrate the scientific method by

- 1 plotting the data
- 2 derive a simple model for the population dynamics
- 3 (fitting parameters in the model to the data)
- 4 using the model predict the evolution other predator-prey systems

Hudson bay data

Year	Hares (x1000)	Lynx (x1000)
1900	30.0	4.0
1901	47.2	6.1
1902	70.2	9.8
1903	77.4	35.2
1904	36.3	59.4
1905	20.6	41.7
1906	18.1	19.0
1907	21.4	13.0
1908	22.0	8.3
1909	25.4	9.1
1910	27.1	7.4
1911	40.3	8.0
1912	57	12.3
1913	76.6	19.5
1914	52.3	45.7
1915	19.5	51.1
1916	11.2	29.7
1917	7.6	15.8
1918	14.6	9.7
1919	16.2	10.1
1920	24.7	8.6

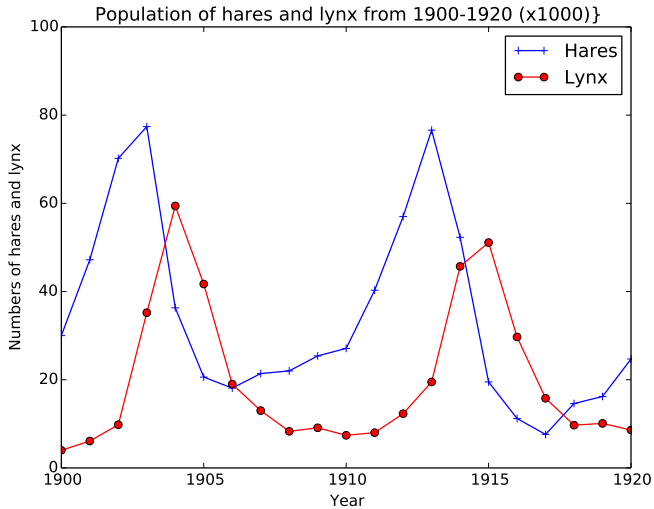
Plotting the data

```
import numpy as np
from matplotlib import pyplot as plt

# Load in data file
data = np.loadtxt('Hudson_Bay.dat', delimiter=',', skiprows=1)
# Make arrays containing x-axis and hares and lynx populations
year = data[:,0]
hares = data[:,1]
lynx = data[:,2]

plt.plot(year, hares, 'b-+', year, lynx, 'r-o')
plt.axis([1900,1920,0, 100.0])
plt.xlabel(r'Year')
plt.ylabel(r'Numbers of hares and lynx ')
plt.legend(('Hares', 'Lynx'), loc='upper right')
plt.title(r'Population of hares and lynx from 1900-1920 (x1000)}')
plt.savefig('Hudson_Bay_data.pdf')
plt.savefig('Hudson_Bay_data.png')
plt.show()
```

Hares and lynx in Hudson bay from 1900 to 1920



Why now create a computer model for the hare and lynx populations?

- We see oscillations in the data
- What causes cycles to slow or speed up?
- What affects the amplitude of the oscillation or do you expect to see the oscillations damp to a stable equilibrium?
- With a model we can better *understand the data*
- More important: we can understand the ecology dynamics of predator-pray populations

The traditional (top-down) approach

The classical way (in all books) is to present the Lotka-Volterra equations:

$$\frac{dH}{dt} = H(a - bL)$$
$$\frac{dL}{dt} = -L(d - cH)$$

Here,

- x is the number of preys
- y the number of predators
- a, b, d, c are parameters

Most books quickly establish the model and then use considerable space on discussing the qualitative properties of this *nonlinear system of ODEs* (which cannot be solved)

The “new” discrete bottom-up approach

The bottom-up approach

- Start with experimental data and discuss the methods which have been used to collect the data, the assumptions, the electronic devices, the aims etc. That is, expose the students to the theory and assumptions behind the data that have been collected and motivate for the scientific method.
- Where appropriate the students should do the experiment(s) needed to collect the data.
- The first programming tasks are to read and visualize the data to see if there are patterns or regularities. This strengthens a research-driven intuition.
- Now we want to increase the understanding through modeling.
- Most of the biology lies in the *derivation* of the model. We shall focus on an intuitive discrete approach that leads to difference equations that can be programmed directly.

The “new” discrete bottom-up approach

The bottom-up approach

- Start with experimental data and discuss the methods which have been used to collect the data, the assumptions, the electronic devices, the aims etc. That is, expose the students to the theory and assumptions behind the data that have been collected and motivate for the scientific method.
- Where appropriate the students should do the experiment(s) needed to collect the data.
- The first programming tasks are to read and visualize the data to see if there are patterns or regularities. This strengthens a research-driven intuition.
- Now we want to increase the understanding through modeling.
- Most of the biology lies in the *derivation* of the model. We shall focus on an intuitive discrete approach that leads to difference equations that can be programmed directly.

The “new” discrete bottom-up approach

The bottom-up approach

- Start with experimental data and discuss the methods which have been used to collect the data, the assumptions, the electronic devices, the aims etc. That is, expose the students to the theory and assumptions behind the data that have been collected and motivate for the scientific method.
- Where appropriate the students should do the experiment(s) needed to collect the data.
- The first programming tasks are to read and visualize the data to see if there are patterns or regularities. This strengthens a research-driven intuition.
- Now we want to increase the understanding through modeling.
- Most of the biology lies in the *derivation* of the model. We shall focus on an intuitive discrete approach that leads to difference equations that can be programmed directly.

The “new” discrete bottom-up approach

The bottom-up approach

- Start with experimental data and discuss the methods which have been used to collect the data, the assumptions, the electronic devices, the aims etc. That is, expose the students to the theory and assumptions behind the data that have been collected and motivate for the scientific method.
- Where appropriate the students should do the experiment(s) needed to collect the data.
- The first programming tasks are to read and visualize the data to see if there are patterns or regularities. This strengthens a research-driven intuition.
- Now we want to increase the understanding through modeling.
- Most of the biology lies in the *derivation* of the model. We shall focus on an intuitive discrete approach that leads to difference equations that can be programmed directly.

The “new” discrete bottom-up approach

The bottom-up approach

- Start with experimental data and discuss the methods which have been used to collect the data, the assumptions, the electronic devices, the aims etc. That is, expose the students to the theory and assumptions behind the data that have been collected and motivate for the scientific method.
- Where appropriate the students should do the experiment(s) needed to collect the data.
- The first programming tasks are to read and visualize the data to see if there are patterns or regularities. This strengthens a research-driven intuition.
- Now we want to increase the understanding through modeling.
- Most of the biology lies in the *derivation* of the model. We shall focus on an intuitive discrete approach that leads to difference equations that can be programmed directly.

The “new” discrete bottom-up approach

The bottom-up approach

- Start with experimental data and discuss the methods which have been used to collect the data, the assumptions, the electronic devices, the aims etc. That is, expose the students to the theory and assumptions behind the data that have been collected and motivate for the scientific method.
- Where appropriate the students should do the experiment(s) needed to collect the data.
- The first programming tasks are to read and visualize the data to see if there are patterns or regularities. This strengthens a research-driven intuition.
- Now we want to increase the understanding through modeling.
- Most of the biology lies in the *derivation* of the model. We shall focus on an intuitive discrete approach that leads to difference equations that can be programmed directly.

Basic (computer-friendly) mathematics notation

- Time points: t_0, t_1, \dots, t_m
- Uniform distribution of time points: $t_n = n\Delta t$
- H^n : population of hares at time t_n
- L^n : population of lynx at time t_n
- We want to model the changes in populations, $\Delta H = H^{n+1} - H^n$ and $\Delta L = L^{n+1} - L^n$ during a general time interval $[t_{n+1}, t_n]$ of length $\Delta t = t_{n+1} - t_n$

Basic dynamics of the population of hares

The population of hares evolves due to births and deaths exactly as a bacteria population:

$$\Delta H = a\Delta t H^n$$

However, hares have an additional loss in the population because they are eaten by lynx. All the hares and lynx can form $H \cdot L$ pairs in total. When such pairs meet during a time interval Δt , there is some small probability that the lynx will eat the hare. So in fraction $b\Delta t HL$, the lynx eat hares. This loss of hares must be accounted for: subtracted in the equation for hares:

$$\Delta H = a\Delta t H^n - b\Delta t H^n L^n$$

Basic dynamics of the population of lynx

We assume that the primary growth for the lynx population depends on sufficient food for raising lynx kittens, which implies an adequate source of nutrients from predation on hares. Thus, the growth of the lynx population does not only depend of how many lynx there are, but on how many hares they can eat. In a time interval ΔtHL hares and lynx can meet, and in a fraction $b\Delta tHL$ the lynx eats the hare. All of this does not contribute to the growth of lynx, again just a fraction of $b\Delta tHL$ that we write as $d\Delta tHL$. In addition, lynx die just as in the population dynamics with one isolated animal population, leading to a loss $-c\Delta tL$.

The accounting of lynx then looks like

$$\Delta L = d\Delta tH^nL^n - c\Delta tL^n$$

Evolution equations

By writing up the definition of ΔH and ΔL , and putting all assumed known terms H^n and L^n on the right-hand side, we have

$$H^{n+1} = H^n + a\Delta t H^n - b\Delta t H^n L^n$$

$$L^{n+1} = L^n + d\Delta t H^n L^n - c\Delta t L^n$$

Note:

- These equations are ready to be implemented!
- But to start, we need H^0 and L^0 (which we can get from the data)
- We also need values for a , b , d , c

Adapt the model to the Hudson Bay case

- As always, models tend to be general - as here, applicable to “all” predator-pray systems
- The critical issue is whether the *interaction* between hares and lynx is sufficiently well modeled by $\text{const}HL$
- The parameters a , b , d , and c must be estimated from data
- Measure time in years
- $t_0 = 1900$, $t_m = 1920$

The program

```
import numpy as np
import matplotlib.pyplot as plt

def solver(m, H0, L0, dt, a, b, c, d, t0):
    """Solve the difference equations for H and L over m years
    with time step dt (measured in years)."""

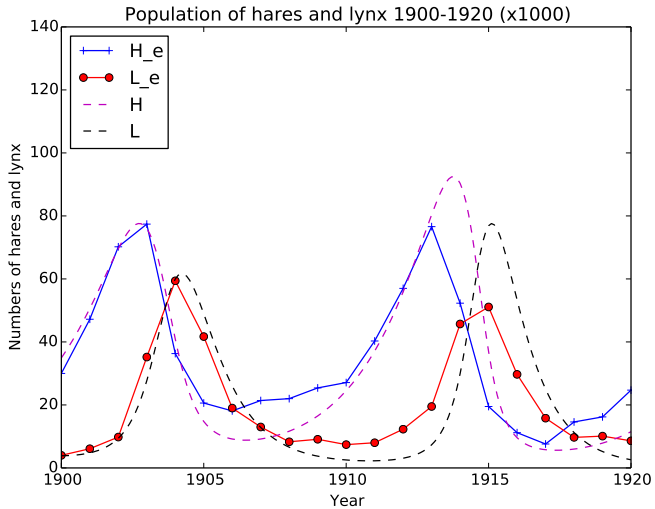
    num_intervals = int(m/float(dt))
    t = np.linspace(t0, t0 + m, num_intervals+1)
    H = np.zeros(t.size)
    L = np.zeros(t.size)

    print 'Init:', H0, L0, dt
    H[0] = H0
    L[0] = L0

    for n in range(0, len(t)-1):
        H[n+1] = H[n] + a*dt*H[n] - b*dt*H[n]*L[n]
        L[n+1] = L[n] + d*dt*H[n]*L[n] - c*dt*L[n]
    return H, L, t

# Load in data file
data = np.loadtxt('Hudson_Bay.dat', delimiter=',', skiprows=1)
# Make arrays containing x-axis and hares and lynx populations
t_e = data[:,0]
H_e = data[:,1]
L_e = data[:,2]
```

The plot



Other examples

- Disease modeling
- Predator-prey with disease
- Bioinformatics: searching in strings
- Move from difference equations to differential equations, would this be meaningful? Probably not - it does not give anything in biology before the models are so complex that one needs other things than Forward Euler...

Example of possible cases, Genomics

- Skal vi si noe om dette ogsaa eller holder det? Her kan vi snakke om feks enkle Markov DNA sekvenser