

Descriptive Account

Biology Students Building Computer Simulations Using StarLogo TNG

V Anne Smith¹ and Ishbel Duncan²

¹*School of Biology and* ²*School of Computer Science, University of St Andrews, UK*

Date received: 28/09/2011

Date accepted: 07/12/2011

Abstract

Confidence is an important issue for biology students in handling computational concepts. This paper describes a practical in which honours-level bioscience students simulate complex animal behaviour using StarLogo TNG, a freely-available graphical programming environment. The practical consists of two sessions, the first of which guides students through building their own computer simulation using StarLogo TNG's graphical programming blocks. The second practical requires them to modify the simulation and carry out a simulation-based experiment. Results from pre- and post-surveys show that, after completing the practical, students have increased their confidence in answering questions requiring understanding of computer code.

Keywords: interdisciplinary, computational biology, complexity science, computer programming

Introduction

The scientific world is increasingly interdisciplinary (Aboelela *et al.*, 2007; Haines *et al.*, 2011). For example, fields such as functional genomics, bioinformatics, and systems biology integrate biology with computational sciences (Bialek and Botstein, 2003; NAP, 2003; APBI, 2008). In this changing scientific landscape, the need to increase quantitative skills of bioscience students has been recognised from the perspectives of both advancement of research science (Bialek and Botstein, 2003; Rocco, 2003) and employability issues of bioscience graduates (NAP, 2003; APBI, 2008). Koenig (2011), in an analysis of quantitative skills training in UK bioscience education, identified problems with confidence — more so than ability — as a barrier in getting bioscience students to apply quantitative methods. Here we describe a computer practical that serves to increase student confidence in a quantitative area.

The computer practical is set in a module concentrating on understanding computational research applied to biology on a conceptual level. The module is Complex Systems in Animal Behaviour, one of a variety of Senior Honours-level modules concentrating on small-group (six to fifteen students), research-topical teaching in the School of Biology at the University of St Andrews. The module introduces students to the ideas behind complexity science (e.g. complex systems, emergence, self-organisation), their application in animal behaviour (e.g. flocking/herding, social insects, collective decision-making), and the tools of complexity science: mathematical models, robotics, and computer programming. The computer practical provides students with the experience of developing a computer simulation, similar to (although simpler than) those they have encountered in reading the primary literature, and analysing simulation results.

The goal of the exercise is to provide hands-on experience with a computer simulation for studying animal behaviour, with the aim of increasing the students' level of confidence in interacting with computational tools. The practical also presents an experiential way to explore and understand complex systems concepts, using the principle of 'constructionism', which

suggests that people learn best by building (Papert and Harel, 1991; Alimisis *et al.*, 2009; Klopfer *et al.*, 2009a; Antonenko and Thompson, 2011). Asking the students to design their own experiment with the simulation provides an enquiry-based educational experience (Hmelo-Silver *et al.*, 2007; Deigan, 2009), increasing student perception of “ownership” of their work (Palmer, 2002).

StarLogo TNG

The main barrier to providing such hands-on, enquiry-led experience to bioscience students in a computational area is the lack of background skills for performing advanced techniques such as building a simulation. However, this barrier can be overcome using StarLogo TNG (The Next Generation), a software tool developed specifically to enable students to build simulations and learn features of complex systems without extensive background knowledge (Klopfer *et al.*, 2009a; 2009b). StarLogo TNG provides an environment for building agent-based models, one of the basic techniques in complexity science studies of animal behaviour. In an agent-based model, individual “agents” (typically an animal) are provided rules of behaviour. Multiple agents following the same rules interact, and thus produce a self-organised emergent property of the system, such as bird flocks (Reynolds, 1987; Hildenbrandt *et al.*, 2010) and ant foraging patterns (Deneubourg *et al.*, 1990). More complex models can allow multiple types of agents to interact, such as forager and nurse bees (Johnson, 2009), a facility also provided by StarLogo TNG.

Instead of text-based computer code, StarLogo TNG has users build simulations using graphical programming blocks. The blocks are coloured based on programming function and “click” together using puzzle-piece shapes that allow only syntactically correct constructs to be built (Klopfer *et al.*, 2009a). This eases learning of basic programming concepts such as variable types, assignment, procedures, and control-flow (e.g. if-then statements, loops) by providing intuitive, visual mappings. For example, understanding control-flow requires little more than visually parsing the *if* or *repeat* commands (Figure 1) to conclude that items placed within the *then* slot will be performed if the items placed within the *test* slot are true, or the items placed within the *do* slot will be subject to the number of repetitions placed within the *times* slot. Additionally, StarLogo TNG’s puzzle-piece shapes avoid one of the main difficulties in learning computer programming, where a non-functioning program is not necessarily the result of conceptual errors in command construction but instead can be due to violation of (arbitrary) syntactical rules such as the visually difficult to distinguish use of a colon instead of a semi-colon. The curved shape of a Boolean variable is easy to distinguish from the pointed shape of a number (Figure 1). StarLogo TNG provides additional feedback with an auditory “click” when syntactically compatible commands are placed together. In this way, students can concentrate on the logical flow of the program without concern about details of syntax.

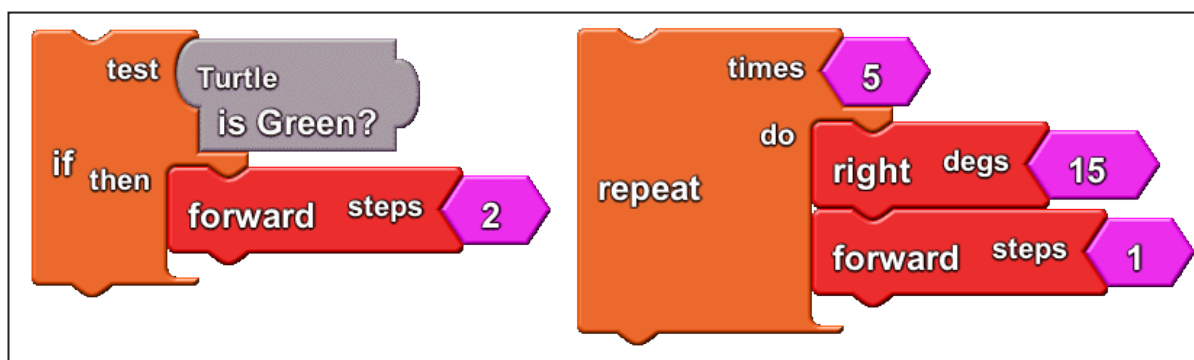


Figure 1 StarLogo TNG’s graphical programming blocks. Example if (left) and repeat (right) blocks are shown. The if block commands a Turtle agent to take two steps forward if it is green; the repeat block commands an agent to repeat five times the sequence of turning right 15 degrees and taking one step forward

StarLogo TNG also provides a 3D visualisation of the simulation output. This is done using high-quality computer graphics with animated characters, meant to be visually appealing to a generation used to 3D rendering in applications such as video games (Klopfer *et al.*, 2009a; 2009b). The running simulation can be viewed through a variety of camera angles, as well as through the point-of-view or over the shoulder of an individual agent. These multiple views provide both entertaining and immediate feedback on the progress of the simulation as it is being built, and enable the student to connect the individual perspective of one actor with the systems-level emergent behaviour observed (Klopfer *et al.*, 2009a).

Finally, StarLogo TNG is freely available for Mac, Windows, and Linux operating systems, making it ideal for educational purposes. It can be installed in as many copies as required on divergent computing platforms for teaching in the classroom. Also, students who wish to explore further can obtain their own copy.

The practical

The computer practical is divided into two sections, performed on separate days. In the first section, students are directed through the process of building a computer simulation in StarLogo TNG. The simulation is one they have seen previously in a lecture, thus the students are aware of how the final product is meant to work. In the second section, they are asked to apply the skills they have learned in the first section by modifying the simulation and running a small experiment. They explore the effect their modification(s) have on the emergent properties of the system.

Simulation building

The students first build an “Ants and Granules” simulation, using simplified rules based on *Leptothorax* ants building nests. These ants live in narrow cracks in rocks and construct a self-organised wall to define the circumference of their nest. In order to build this wall at a new nesting site, wall-building ants wander randomly until they encounter a granule of building material (e.g. grain of sand or a dirt particle). They then return to the edge of the main mass of newly migrated ants, turn around, and push the building material outward until either they encounter resistance (i.e. the growing wall) or a certain time has elapsed. These simple rules result in the production of a nest of appropriate size for the colony (Franks *et al.*, 1992). The Ants and Granules simulation further simplifies these rules to remove the central mass of ants and the time restriction, leaving two rules: (1) ants wander until they encounter a granule, at which point they pick it up and (2) once carrying a granule, ants wander until they encounter another granule, where they drop the granule they are carrying. Instead of a wall, the simplified simulation creates an emergent property of granules gathering in clumps (incidentally, making it more similar to the initiation of columns in termite mounds (Bonabeau *et al.*, 1998); Figure 2).

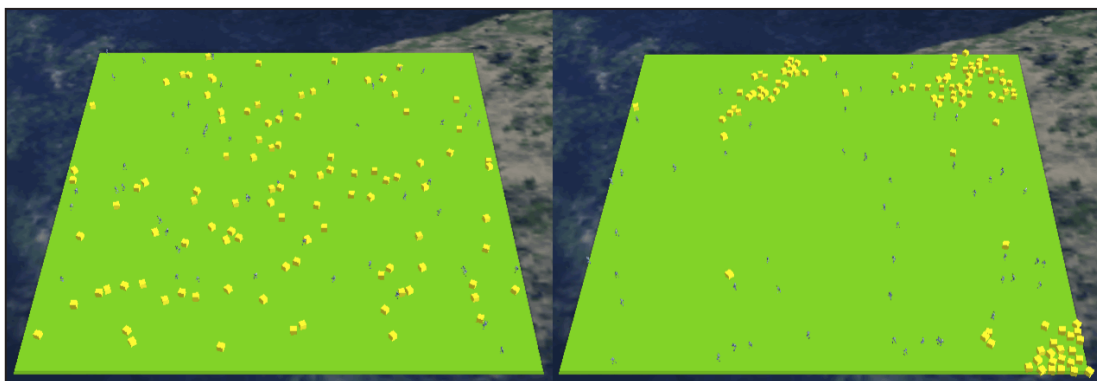


Figure 2 Emergent property in the ‘Ants and Granules’ simulation. An initial random placement of Granule agents (left) is sorted into clumps (right) by the self-organised behaviour of the Ant agents

A handout guides the students through a step-wise, debugging process of programming this simulation. Key to this section of the practical is that the handout does not provide a “cook-book” set of instructions, but instead mimics the analyse-build-test process through which many programmers develop code: students are directed often to view the output of their simulation so far, consider what might be causing the results they see, modify the program to fix any undesired behaviour, and only then move on to creating next feature of the simulation. This process is necessarily non-linear, involving discarding blocks of program in order to replace them with better-performing blocks, which are analysed and tested in turn.

For example, Figure 3 shows two excerpts of the handout. The first excerpt (top) starts just after the students have been instructed on how to create *Ants* and *Granules* within a setup procedure. It instructs them to view what they’ve done, where they will see that the default behaviour is to create agents in the centre of the simulation world. As this is not the desired behaviour (which they know from having viewed a working simulation previously), it then directs them in inserting the appropriate command to scatter agents throughout the world. Students are directed again to look at the output, which should now create the desired scattering of agents. Only then is the next refinement, uniformly colouring the agents, introduced. The next excerpt (bottom) picks up after they have pieced together the commands to set agent colour, but have not yet included them in the *setup* procedure. To do this, the students need to discard code they designed previously and replace it with the new blocks. Once this is done, they are again instructed to view the output. This time, another undesired behaviour will become evident: as no reset-type command has been introduced, the new uniformly coloured agents simply fill the world already occupied by the old agents. It may take the students several iterations of running their *setup* procedure before this becomes obvious. At that point, they are then directed in how to fix this new problem.

B. Let’s see what we’ve done so far. Go to the [SpaceLand](#) window. Click on *setup*. What happens now?

We’ve made 50 *Ants* and 100 *Granules*, but they are all standing on top of each other in the centre of the land. We’d rather have them scattered throughout. Go back to [StarLogoBlocks](#).

- Return to the *Factory* (click on arrow to left of *My Blocks*, labelled “*Factory*”).
- Click on *Setup and Run* and choose from the drawer the command *scatter everyone*.
- Click this in below *create Granules*.

C. Go to [SpaceLand](#) and click on *setup*. Now what happens?

Notice how the granules are all different colours. Let’s make them all be the same colour. Go to [StarLogoBlocks](#).

Now we’ve made a command that will create 100 *Granules* and turn them all one colour.

- Click in the new *create Granules (num, do)* below *create Ants* where the old *create Granules (num)* was before.
- Move *scatter everyone* away from the old *create Granules (num)* and click it in below the new *create Granules (num, do)*.
- Clean up by moving the old *create Granules (num)* to the trash bin.

D. Go to [SpaceLand](#) and click *setup*. What happens? Click *setup* a couple more times. Now can you see what is happening?

When we click *setup*, what we’d really like is everything to start from fresh. So we need to put that into the commands. Return to [StarLogoBlocks](#).

Figure 3 Two excerpts from the practical handout

In this way, students engage in the process of building the simulation. Simultaneously, they are exposed to the trial-and-error nature of computer programming. This should succeed in diffusing any misconceptions of a simulation as a monolithic, mysterious creation by presenting computer programming as a task manageable in small chunks. The initial formulation of the practical only allowed one hour’s time for both sections. This proved to be too short, especially

for the first section; only a small number of students completed the entire simulation. Future iterations of the practical allowed two hours for both sections, and this time period was sufficient for all students to complete the simulations. Speedier students were observed exploring other possibilities inherent in StarLogo TNG once the simulation was completed.

Simulation modification

The second section of the practical directs students to modify one or two elements of the simulation and perform experiments to evaluate the modifications' effects on the emergent properties. A few example possibilities of modifications are provided, and the students are invited to invent their own. These modifications require understanding of the computer program and how changing the commands results in changing the simulation. It also serves to exercise student creativity and allows them to explore possibilities in an enquiry-based manner.

Many students explore modifying the number of Ants, Granules, or their ratio; these modifications shows one of the largest effects on the size, speed, and existence of Granule clumps (e.g. with more Ants than Granules, eventually all Granules are being carried and none are ever dropped). Other modifications influence parameters of the wandering behaviour of Ants, behaviour of Granules once they are dropped, and similar items. Students have also presented modifications with unexpected effects. For example, student experiments have shown how changing the animated characters used in the simulation influences the emergent clumping behaviour, because it modifies the size of the items doing the colliding. Some modifications performed by students require additional programming and problem-solving. For example, another common modification is to change the initial spatial distribution of Ants or Granules. Because this was handled in the original simulation with one *scatter everyone* command, the production of an effect like all granules in a line at the edge of the world requires: (1) discovery of the ability to set the (x,y) coordinates of agents, (2) trial and error to determine that the coordinate (0,0) represents the centre of the world and that x and y both range from -50 to 50, and (3) use of multiple mathematical functions to produce a random scattering of values spanning the extent of one axis.

Finally the students write a small report, in the style of a research paper, on the outcome of their simulation experiment. These reports serve to consolidate the learning of complex systems concepts, by requiring the students to explain the resulting changes in emergent properties based on the changes they made in individual agents. The reports also serve to complete the cycle of ownership of their enquiry-based project, where they are able to explain, analyse and reflect on their choices.

Influence on student confidence

As part of an on-going research project assessing the effect of this practical on students' perceptions of computer programming, students are presented with surveys which include problem-solving questions requiring understanding computer code. A pre-survey is presented before the first practical session. A post-survey containing identical problem-solving questions is presented after the second practical session. In these surveys, students rate their confidence in their answers to the problem-solving questions.

The problem-solving questions were presented as multiple-choice, but always with the option of a "write-in" answer if the student could not find their desired answer among the given choices. There were three types of problem-solving questions; value-, word-, and StarLogo-based (Figure 4). In the value-based questions, students were presented with textual code (written in C-like syntax) performing value assignments and addition operations to variables *a*, *b*, and *c*. The students were prompted to pick from choices showing the new values of modified variables. In the word-based questions, the students were presented with textual

code (C-like) showing print statements of the words *hello* and *bye* controlled by the control-flow elements of if-then statements, loops, and procedure calls. They were prompted to pick from choices showing the pattern of resulting printout of *hello*s and *bye*s. In the StarLogo-based questions, the students were presented with graphical command blocks from StarLogo TNG, again controlled by control-flow elements of if-then statements, loops, and procedure calls. They were prompted to pick from choices showing the path followed by an agent under these commands. Students were asked to rate their confidence in their provided answer on a five-point Likert scale ranging from low to high confidence.

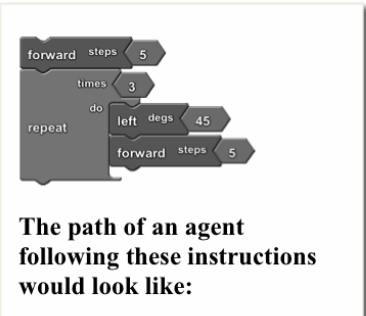
value-based	word-based	StarLogo-based
<pre>int a = 10; a = a + 1;</pre> <p>The new value of a is:</p>	<pre>int a = 10; if (a < 20) then {print "hello "}; else {print "bye "};</pre> <p>The printout would read:</p>	 <p>The path of an agent following these instructions would look like:</p>

Figure 4 Examples of the three question types on pre- and post-surveys. Each question type was followed by multiple-choice presentation of potential answers (including the correct one); a “write in” option was also provided

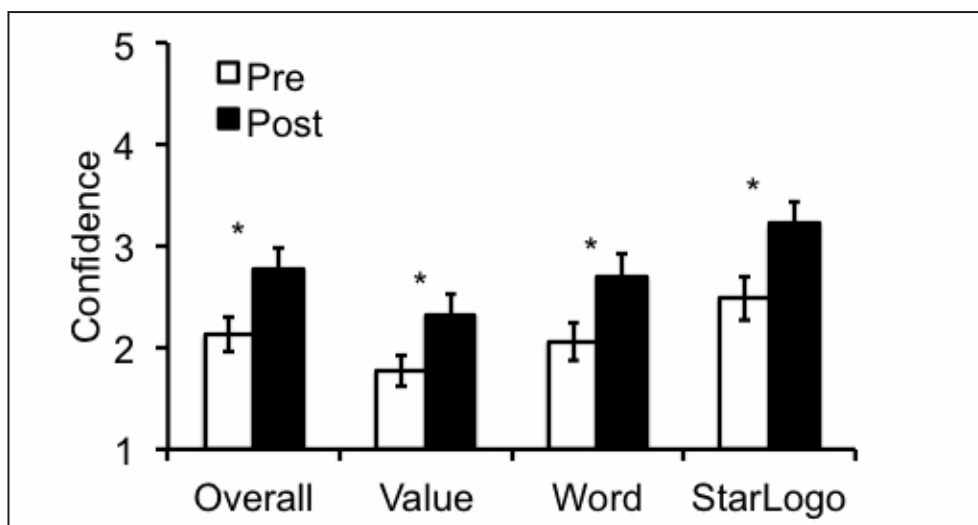


Figure 5 Results from pre- and post-surveys. Confidence (scale: 1 = low; 5 = high) on answered questions overall and for each question type before (Pre: open bars) and after (Post: filled bars) the practical. Mean ± SE plotted for all. * – significant paired t-test (values in text)

Both pre- and post-surveys were presented to 38 students comprising two cohorts at the University of St Andrews performing the described practical (2009 and 2010) and one at Anglia Ruskin University performing a very similar practical (2010). Compared to their confidence prior to the practical, student confidence after the practical increased on all question types ($t = -3.6$, $df = 37$, $p \leq 0.001$ for all, Figure 5). This demonstrated that the practical succeeded in increasing confidence in these computational tasks. Students provided positive verbal feedback during and after the exercise, and were observed to continue exploring the StarLogo TNG program after completion of the required elements. Several students indicated that they had downloaded StarLogo TNG to their personal computers and were planning to continue

to use it. Thus it appears that the practical has had a continued positive effect on students' attitude towards using computational tools.

Additionally, while complete analysis of the accuracy of the students' answers must await the larger sample sizes of the completed study, the survey results already show that after the practical, students were more accurate on StarLogo-based questions (paired t-test, $t = -3.1$, $df = 37$, $p = 0.004$), although the data yet shows no difference overall or on other question types ($t = -1.4$, $df = 37$, $P \geq 0.16$ for all). The increase in accuracy on StarLogo-based questions shows that the students gained knowledge of programming specific to the StarLogo TNG environment, which includes understanding basic concepts like variables and control-flow. Evaluating the effect on accuracy was complicated by the ceiling effect, where higher accuracy on the pre-survey limited the amount of potential increase in the post-survey. In this case, the StarLogo-based questions both were the object of direct experience in the practical and had the most room for improvement in the survey, for 49%, 69%, and 38% of students accurately answered more than half of the questions in the pre-survey for the value-, word-, and StarLogo-based questions, respectively. Thus, more data is needed to determine whether the skills learned are limited to StarLogo TNG or whether they might extend to other types of programming. This question will continue to be explored as part of the on-going research project.

Discussion

The practical described in this report could potentially serve several educational purposes. In this module, it was used both to enable students to understand the process of simulation-based complexity research without having to first learn programming and to increase student confidence with such computational tools. Thus, it can function as a "light-touch" foray through computation-intensive research areas, providing conceptual understanding of how the research is accomplished. Another possible use of this practical could be as a starting point for learning programming. Presenting quantitative concepts within a biological context increases their relevance to bioscience students and motivates their learning (Koenig, 2011). The graphical programming blocks of StarLogo TNG allow students to begin programming immediately, without a precursor of abstract-level information about the features of computer programs. Following up on this practical with further programming tasks in StarLogo TNG, and eventual relation to more traditional text-based code, would embed the programming within its biological context from the start, and thus enhance students' motivation for obtaining the skill.

Beyond this specific practical, the StarLogo TNG environment could serve wider educational purposes across biology, because it allows students without a computational background to engage with computational models. The practical described here was used to teach concepts of agent-based modelling and emergent behaviour found in complexity approaches to animal behaviour. Other areas of the biosciences also lend themselves to StarLogo TNG-based practical exercises. For example, ecology and epidemiology use agent-based models for exploring population dynamics and disease spread. Using StarLogo TNG in the teaching of such concepts would allow the level of computational engagement to be modified to an instructor's educational desires. Presenting students with a completed model in which they modified features using the 3D interface would serve to teach concepts of the system only; instructing students in building their own simulation, as presented here, would serve to provide a structured approach to the students' experiential exposure to modelling.

The survey results demonstrating the overall increase in student confidence in answering the various computer code-based questions is encouraging for the ability of this, or similar practicals, to break down the conceptual divide (Koenig, 2011) between bioscience and quantitative subjects in students' minds. Manipulation of complex data sets and consequent computational tools are a requirement of modern-day science in all fields (Lazer *et al.*, 2009).

Bioscience graduates having familiarity with computational research is positive for the continued integration of interdisciplinary research. This direct experience with computational methods would tend to enhance collaboration and willingness to consider complementary approaches. Thus, exercises like this practical, which serve to increase student confidence, will be beneficial to the education of bioscience students in this increasingly more quantitative world.

Resources

StarLogo TNG is available for download from MIT's Schneller Teacher Education Program (MIT STEP, 2010). The practical handout and completed simulation are available by request and on the corresponding author's website (Smith, 2011).

Acknowledgements

We are grateful to Dr Toby Carter of Anglia Ruskin University for carrying out our pre- and post-survey with students in his module.

Corresponding Author:

V Anne Smith, Sir Harold Mitchell Building, School of Biology, University of St Andrews, St Andrews, Fife, KY16 9TH
Tel: 01334 463 368; Email: anne.smith@st-andrews.ac.uk

References

- Aboelela, S. W., Larson, E., Bakken, S., Carrasquillo, O., Formicola, A., Glied, S. A., Haas, J. and Gebbie, K. M. (2007) Defining interdisciplinary research: Conclusions from a critical review of the literature. *Health Services Research* **42** (1–1), 329–346
- Alimisis, D., Frangou, S. and Papanikolaou, K. (2009) A constructivist methodology for teacher training in educational robotics: the TERECoP course in Greece through trainees' eyes. In *Proceedings of the Ninth IEEE International Conference on Advanced Learning Technologies, 2009, Riga, Latvia*, eds Aedo I., Chen, N. S., Kinshuk, Sampson, D. and Zaitseva, L., pp. 24–28
- Antonenko, P. D. and Thompson, A. D. (2011) Preservice teachers' perspectives on the definition and assessment of creativity and the role of web design in developing creative potential. *Education and Information Technologies* **16**, 203–224
- APBI (2008) *Skills Needed for Biomedical Research: Creating the Pool of Talent to Win the Innovation Race*. London, UK: Association of the British Pharmaceutical Industry, available at <http://www.abpi.org.uk/our-work/library/industry/Pages/skills-biomedical-research.aspx> (Accessed 8 Sep 2011)
- Bialek, W. and Botstein, D. (2003) Introductory science and mathematics education for 21st-century biologists. *Science* **303**, 788–790
- Bonabeau, E., Theraulaz, G., Deneubourg, J. L., Franks, N.R., Rafelsberger, O., Joly, J. L. and Blanco, S. (1998) A model for the emergence of pillars, walls and royal chambers in termite nests. *Philosophical Transactions of the Royal Society of London, Series B Biological Sciences* **355**, 1561–1576
- Deignan, T. (2009) Enquiry-Based Learning: perspectives on practice. *Teaching in Higher Education* **14** (1), 13–28
- Deneubourg, J. L., Aron, S., Goss, S. and Pasteels, J. M. (1990) The self-organizing exploratory pattern of the argentine ant. *Journal of Insect Behavior* **3**, 159–168
- Franks, N., Wilby, A., Silverman, W. W. and Tofts, C. (1992) Self-organizing nest construction in ants: sophisticated building by blind bulldozing. *Animal Behaviour* **44**, 357–375
- Haines, V. A., Godley, J. and Hawe, P. (2011) Understanding interdisciplinary collaborations as social networks. *American Journal of Community Psychology* **47**, 1–11
- Hmelo-Silver, C. E., Duncan, R. G. and Chinn, C. (2007) Scaffolding and achievement in Problem-Based and Inquiry Learning: A response to Kirschner, Sweller, and Clark (2006) *Educational Psychologist* **42** (2), 99–107

- Hildenbrandt, H., Carere C. and Hemelrijk, C. K. (2010) Self-organized aerial displays of thousands of starlings: a model. *Behavioral Ecology* **21**, 1349–1359
- Johnson, B. R. (2009) Pattern formation on the combs of honeybees: increasing fitness by coupling self-organization with templates. *Proceedings of the Royal Society Series B Biological Sciences* **276**, 255–261
- Klopfer, E., Scheintaub, H., Huang, W. and Wendel, D. (2009a) StarLogo TNG: Making agent based modeling accessible and appealing to novices. In *Artificial Life Models in Software*, eds Komosinski M. and Adamatzky, A, pp155–182. London, UK: Springer-Verlag
- Klopfer, E., Scheintaub, H., Huang, W, Wendel, D. and Roque, R. (2009b) The simulation cycle — combining games, simulations, engineering and science using StarLogo TNG. *Journal of E-Learning and Digital Media* **6** (1), 71–96
- Koenig, J. (2011) *A survey of the mathematics landscape within bioscience undergraduate and postgraduate UK higher education*. Leeds, UK: UK Centre for Bioscience
- Lazer, D., Pentland, A., Adamic, L., Aral, S., Barabási, A. L., Brewer, L, Christakis, N., Contractor, N., Fowler, J., Gutmann, M., Jebara, T., King, G., Macy, M., Roy, D. and Van Alstyne, M. (2009) Computational social science. *Science* **323**, 721–723
- MIT STEP (2010) *StarLogo TNG | MIT STEP*. <http://education.mit.edu/projects/starlogo-tng> (Accessed 8 Sep 2011)
- NAP (2003) *BIO2010: Transforming Undergraduate Education for Future Research Biologists*. Committee on Undergraduate Biology Education to Prepare Research Scientists for the 21st Century, Board on Life Sciences and Division on Earth and Life Studies, National Research Council. Washington, DC, USA: The National Academies Press, available at <http://www.nap.edu/catalog/10497.html> (Accessed 8 Sep 2011)
- Palmer, S. (2002) Enquiry-based learning can maximise a student's potential. *Psychology Learning and Teaching* **2** (2), 82–86
- Papert, S. and Harel, I. (1991) Situation constructionism. In *Constructionism*, eds Harel, I. and Papert, S., pp1–12, Norwood, NJ, USA: Ablex Publishing, available at <http://www.papert.org/articles/SituatingConstructionism.html> (Accessed 23 Sep 2011)
- Reynolds, C. W. (1987) Flocks, herds and schools: A distributed behavioral model. *Computer Graphics* **21**, 25–34
- Rocco, M. C. (2003) Converging science and technology at the nanoscale: opportunities for education and training. *Nature Biotechnology* **21** (10), 1247–1249
- Smith, V. A. (2011) *V. Anne Smith: Downloads*. <http://biology.st-andrews.ac.uk/vannesmithlab/downloads.html> (Accessed 8 Sep 2011)