Can Evolution Make New Biological Software?

Richard W. Stevens*

Abstract

The modern theory of biological evolution focuses almost entirely on how random mutation (including recombination) with natural selection could produce all of the physical features and functions that appear in plants and animals. Yet, how do new species of animals obtain the knowledge (the software) to operate the new limbs, organs, and other features (the hardware) that evolution produces? This article will look at reptile-bird evolution and the need for the simultaneous evolution of flying software to support the evolution of the flying hardware (the wings). Next, the article addresses whether random mutation and natural selection can modify software to achieve any new useful functions. Using the InforMutation simulation system, the analysis shows that random modifications to computer software yield broken programs, not improved software. By analogy, biological control systems cannot be modified randomly and then be able to operate new biological hardware.

Introduction

The chief historical evidence of evolution is allegedly the fossil record. The fossil record shows the vast array of plant and animal types that existed in the past, but fossils can provide direct evidence only about general body shapes, bone configurations, and external forms. The fossil record does not indicate how animals, for example, obtained the *information* (knowledge or skill) to operate their organs, limbs, and other features. Yet unquestionably all animals possess enormous amounts of such operational information (Devlin, 2005; Behe, 1996). Richard Dawkins has claimed that undirected evolution can produce the information necessary to develop and operate living things (e.g. Dawkins, 1986, pp. 47–48). Many writers have argued convincingly that Dawkins is wrong (Behe, 1996; Bergman, 2001; Dembski, 2002; Poppe, 2005; and Spetner, 1999).

Evolution posits that all extant species descended from previous species that underwent a long series of beneficial mutations that caused species A to give rise to species B, and species B to give rise to species C, and continuing. *For purposes of argument only*, this article assumes neo-Darwinian processes of common descent can create new species with new physical features. This article focuses on a distinctly different issue: Can evolutionary processes modify biological operating instructions to control and operate the new features?

Biological operating instructions are logically analogous to computer operating instructions. Evolutionists have expressly drawn this analogy (e.g. Levitin, 2006, pp. 118, 131, 169, 170; Lenski, 2003). For example, Daniel J. Levitin wrote, "as ... with memory, computer models can help us grasp the inner work-

* Richard W. Stevens, 4541 E. Vista Drive, Phoenix, AZ 85032, Tstevens96@cox.net

Accepted for publication September 21, 2008

ings of the brain" (Levitin, 2006, p. 169) and "the brain represents all ... aspects of the world in terms of mental or neural codes" (Levitin, 2006, p. 177). Dawkins (1995) wrote that genes contain

> within their minute internal structure, long strings of pure digital information [that] are truly digital, in the full and strong sense of computers and compact disks. The genetic code is not a binary code as in computers ... but a quaternary code, with four symbols. The machine code of the genes is uncannily computerlike. Apart from differences in jargon, the pages of a molecular-biology journal might be interchanged with those of a computer-engineering journal. (p.17)

Accordingly, to evaluate whether evolutionary processes can modify biological operating instructions, it is reasonable to test whether those same types of processes can modify computer program instructions. InforMutation (Stevens, 2007), an Internet-based simulation system described in the latter sections of this article, allows the user to evaluate whether evolutionary processes are capable of making successful changes to a computer program. The simulation tests whether it is possible to produce a series of favorable mutations in a computer program's operating instructions, this process being analogous to mutating the operating instructions that a species requires to use a new limb, organ, or other feature. (Evolutionist researchers have expressly adopted the validity of analogies drawn from computer-simulated mutation and selection; e.g. Lenski, 2003.)

By direct analogy, if evolution-like processes cannot successfully modify computer programs, then it is unlikely that those same kinds of processes could successfully modify the operating instructions that control organs, limbs and other animal features (Appendix I summarizes the argument presented in this article).

How Birds Evolved from Reptiles

Neo-Darwinism asserts every species of plant and animal has emerged by the process of biological evolution (e.g. Van Biema, 2006, pp. 49–55; Kutschera and Niklas, 2004, pp. 256, 259). Evolution occurs without purpose by the dual materialistic mechanisms of *random mutation* and *natural selection* (Kutschera and Niklas, 2004).

Random mutation

Many evolutionary theorists claim birds evolved from reptiles (Southwood, 2003; Hecht, 2000). Their claim rests on these assumptions: (1) that random mutations in the genetic structures of reptiles caused the reptile-like creatures to develop wings, and (2) that the mutations were heritable. The mutations leading to the growing of wings did not occur all at once (Dial et al., 2006; Damuth, 2001). Rather, each mutation produced small changes in the reptiles' genome that could be passed from the parents to their offspring, and each subsequent genetic change over time caused the reptilian species' bodies to change. Over the generations the mutations continued to occur and accumulate, resulting in more changes in the reptiles' bodies (Kutschera and Niklas, 2004; Pinker, 1997).

Evolutionists claim reptiles eventually developed wings from existing front feet (Clark, 2007). No recovered fossil record can show exactly how the early pre-bird reptiles manifested their new birdlike features.

Natural selection

Natural selection occurs as the environment acts on the living organisms, such that:

- organisms that are better fit to survive and reproduce in the local environment do in fact survive and reproduce in larger numbers
- organisms less suited tend to die out or reproduce in smaller numbers (Gould, 1996; Kutschera and Niklas, 2004).

A typical evolutionary scenario claims to explain the purely physical changes that might transform saltwater fish species into freshwater species. In a given body of water, the fish species carry genetic differences that make some individuals more tolerant of low salinity than other individuals. As environmental events decrease the salinity of a body of water, the fish able to tolerate the change will survive and reproduce in greater numbers than their less tolerant brethren. Natural selection occurs when the low-salt-tolerant fish survive and reproduce in greater numbers—their tolerance has conferred upon them a survival advantage. As the salinity of the water decreases further, the pressure of natural selection continues to favor the low-salt-tolerant varieties, while the other varieties die out. By these events saltwater fish evolve into freshwater fish (Pinker, 1997; Carroll, 2006). By favoring species with survival advantages (such as toleration of salinity changes), natural selection is considered the evolutionary force that creates new animal species from previous ones (Mlot, 1997).

Evolutionary theory postulates that the same kind of process on a greatly expanded scale applies to the reptile-to-bird scenario. As the pre-bird reptiles undergo mutations, they begin to form wings and structures to support and operate the wings. These mutations would somehow confer survival advantages to the mutants. Natural selection thus would favor the reptiles with stubby or incompletely formed wings (Dial, 2006), so that these pre-bird reptiles could survive and reproduce in greater numbers than the reptiles lacking pre-bird features (Pinker, 1997; Southwood, 2003). Over time, the pre-bird reptiles accumulate more mutations that confer to pre-bird reptiles some survival advantages over the non-bird reptiles. Evolution theory thus posits that, after millions of years and millions of generations of pre-bird reptiles undergoing these physical changes, evolution produces a bird with wings.

A Missing Ingredient Is Evolved Knowledge

Assuming reptile species might have evolved as described, i.e., with physical structures becoming wings and reptile scales or hair becoming feathers (Padian, 2001; Clark, 2007; Hecht, 2000), there remains a crucial but unanswered question: Will this reptile-turned-bird *know how to fly?* This question cannot be dismissed by asserting "the ability to fly must also have evolved while the wing evolved." Evolution needs to offer a reasonable explanation of the mutation-by-mutation steps (each one subject to natural selection) that account for the origin of the bird's *knowledge and ability* to fly.

The saltwater-to-freshwater fish evolution scenario involved changes to physical equipment only. That scenario did not require changes in operation or behavior of the new fish species. In reptile-to-bird evolution, however, there must be changes to physical equipment (producing wings) along with changes enabling the reptile to operate the new equipment (fly) in a quite different, three-dimensional environment, i.e., the troposphere.

To prosper under natural selection, the wings must provide the bird with a survival advantage over the reptile. A (former reptile) bird might have all of the physical equipment to fly, but that physical equipment is useless for flying if the bird does not know how to fly. A nonflying bird in a reptile world is only a reptile with extra baggage. The wings cannot confer the survival advantage of flying if the bird cannot fly.

If flying confers no survival advantage, then natural selection will not favor the reptile-birds who can fly. Recalling the fish example, if low-salt-tolerant varieties of fish lack a survival advantage, then they will not out-reproduce the other varieties. Natural selection will not favor varieties of more or less salt-tolerant fish unless such tolerance confers a survival advantage. Similarly, if flying is not a survival advantage, then natural selection cannot favor the flying reptiles over the ground-based reptiles. This case thus shows that a new physical feature cannot be favored by natural selection unless the feature is immediately functional, and for wings that requires concurrent knowledge of flying.

Natural selection can only select a trait that already exists in some measure in the population (Pinker, 1997). For natural selection to favor reptile-birds who have better flying knowledge and skills, the reptile-bird population must already possess some existing flying knowledge and skills (Ridley, 2002). Evolutionary theory must conclude that flying conferred a survival advantage to the reptiles-turned-birds. Yet the original question persists: how did the reptile-birds obtain the knowledge to fly? Three-dimensional movement in the troposphere would require an unprecedented and untested means of mobility and conscious perspective. There is no reason to assume that a reptile has any of the mental functions or aptitudes necessary for the challenges of flight.

Whether Walking Converts to Flying

Operating a wing is somewhat like operating a leg: moving the leg back and forth produces motion, and moving the wing back and forth produces vertical movement. There is nothing simple about walking (ex. Devlin, 2005, p. 43), but let us assume that a walking motion can convert directly to actuating wings. The pre-bird reptiles had stubby winglike legs, and when they ran, they lifted off the ground (Southwood, 2003, pp. 159–160, 163).

By what evolutionary mechanism could the accidental effect of lifting off the ground—caused by the presence of winglike legs—become converted into brains *programmed from birth* to execute the intricate, specialized, and highly diverse flying skills of a hawk, sea gull, swallow, or hummingbird? Moving a leg or wing does not cause reprogramming of the brain or the changes in the DNA necessary for such a reprogramming. Somehow the evolving bird's brain (to include its nervous system) must be programmed, like a computer, to recognize and use the advantages of wings.

Can Gliding Convert to Flying?

Consider the possibility that the pre-bird reptiles gained an advantage from having their stubby winglike structures by using them to glide down from trees (Hayden, 1999; Kutschera, and Niklas, 2004, p. 263). The possibility that the pre-wings might be used for gliding does not explain if, how, or when the pre-bird reptiles knew how to use this gliding feature.

Perhaps such a pre-bird reptile jumped from a tree to avoid a predator and glided to safety. What feature of the reptile's brain and nervous system, however, would record that accident and be able to repeat that gliding maneuver? More to the point, by what mechanism would the experience of an accidental glide in one (or even several) stubby-winged reptiles become *converted* into built-in brain software by which a future pre-bird or bird would know how to fly (or even just purposefully glide)?

Modified Software Must Coincide with Physical Evolution

These questions highlight a key problem for evolution theory: it does not explain how the knowledge (e.g., brain, neural, or cellular software) to operate a new feature comes into existence. Yet the two developments—the feature and its operational system—must happen nearly simultaneously. Evolutionists assert learned knowledge is not inherited by the young animal from its parents (e.g., Gould, 1996, p. 222; Pinker, 1997, pp. 158–159; Kutschera and Niklas, 2004, p. 260). It follows that knowledge or intelligence is either hardwired into the animal's body (brain or cells) or learned by the individual animal (Greenspan, 2002, pp. 595–596, 607; Pinker, 1994, p. 18). To operate a pair of wings requires the biological equivalent of a computer. Hardware is not enough; to fly with wings, the bird's brain needs the software, i.e., the set of instructions that operate the computer that controls the wings to accomplish flying (Pinker, 1997, pp. 10–12).

Feasibility of Mutating Software

The transformation of a grounded reptile into an accomplished flying bird requires numerous beneficial mutations to develop the wing bones, muscles, and feathers, as well as a set of control instructions. The control instructions are analogous to computer software (Pinker, 1994). Such an analogy requires us to consider whether it is even possible, subject to natural selection, to mutate software to make it "better" or at least different in a way that functionally deploys the new wing feature.

Software is required in the reptile's brain and body to enable it to walk; different software is required to enable flight. Every change in size or shape of a limb or wing would require changes in software. For evolutionary predictions to account for any new features in any new species, the theory must concretely explain how evolutionary processes can *modify* the necessary control software.

InforMutation Simulation Shows Effects of Mutation Upon Software

The InforMutation system (Stevens, 2007) is a simulation that enables the user to explore how mutations affect software. The user can test whether it is possible for a series of mutations to change software and thereby create new and interesting programs. The user can see the actual instructions, instigate random changes to the instructions, and then witness how the computer executes the instructions and produces the results.

InforMutation runs a software program that calculates and prints out all of the prime numbers from 2 to 101. The late astronomer and ardent evolutionist, Professor Carl Sagan, considered the production of a long series of prime numbers (in ascending order) a task that requires advanced intelligence. The 1997 movie *Contact*, based on Sagan's book (Sagan, 1997, pp. 68–69), used this series of primes as strong evidence of intelligent life outside our solar system.

The InforMutation user can cause random mutations in the prime number generator program. The mutations include single-bit and single-character changes or exchanges, as well as multiple-bit and multiple-character changes or exchanges. These "mutations" resemble genetic mutations, e.g., nucleotide substitutions, insertions, deletions, or rearrangements (Sniegowski, 2002). The user can observe the mutations and watch the computer attempt to run the mutated software.

By changing the number and types of mutations, the InforMutation user can observe in real time whether mutations tend to improve, damage, or have little effect on software. Experience with InforMutation has shown most of the software mutations damage the software so that the mutated prime number program either does not deliver the correct (or "improved") number series, or does not run at all.

"But Evolution Does Not Promise Any Improvements"

Gould emphasized that evolution has no purpose or direction: "The basic theory of natural selection offers no statement about general progress, and supplies no mechanism whereby overall advance might be expected" (Gould, 1996, p. 136). Pinker applied this to the evolution of intelligence: "Natural selection does nothing even close to striving for intelligence" (Pinker, 1997, p. 153). Yet without intelligence to guide modifications to software, evolution theory implicitly still assumes the software must somehow evolve.

What would constitute an evolutionary improvement in software? In particular here, what changes (mutations) in InforMutation's prime number program would be considered "improvements" or beneficial to the program?

The InforMutation system allows the user to evaluate whether changes to the prime number program could "mutate" it into a program that does something completely different—in Pinker's words, something "interesting"—and does it correctly (Pinker, 1997, p. 162). Interesting new programs that the mutation process might hypothetically create could include ones that produce:

(1) the odd numbers from 1 to 101;
(2) the even numbers from 2 to 100; or
(3) the prime numbers from 2 to 1001.

To test the effects of much larger changes that are closer to the magnitude of changes needed to go from operating a leg to operating a wing, the "goals" of running InforMutation could include evolving a program that:

(1) prints all 26 letters from A to Z in order;
(2) receives a number from the user and prints out the same number times two;
(3) plays blackjack with the user; or
(4) draws a picture of an animal.

InforMutation can be used to estimate how many mutations would be needed to convert the prime number program to accomplish any one of these possible new functions. To date, InforMutation experiments have shown *no* likelihood that random mutation could do anything more than damage the program.

Software Failure = Disability or Death

Natural selection operates by favoring the increased reproduction rates of the more fit members of a given species. Analogously, in the InforMutation system the failure of the program to run is the equivalent of disability or early death in the natural environment—the program does not function and reproduce. Table I lists the most common program defects and output variations that random mutations cause within the InforMutation system.

Software program failure analogizes directly to the winged reptile's predicament when the reptile lacks properly functioning biological software. For the reptile to become a bird, an unbroken series of successful random (undirected) mutations must occur in both the hardware and the software. If the reptile's software has mutated to a nonfunctional state, or if the wings function poorly or not at all, then the whole process is terminated, since the nonfunctional software will have no second chance to get it right.

When the bird finally emerges, it has to operate its wings to gain the survival advantage that wings can provide. If the (former) reptile's software has been mutated and no longer functions effectively, then the wings will function poorly or not at all. If the reptile with this mutant software reproduces, then presumably the offspring reptiles will inherit the ineffective software. If the reptile's software is so faulty that the reptile dies or is killed prematurely, then the mutated software will never get another opportunity to further mutate.

InforMutation concretely illustrates the practical improbability that the simulation, even run thousands of times, would ever deliver an unbroken series of mutations, all of which functioned well enough not to crash the program and would convert the program from a prime number generator into some other functional program. The improbability of making a mutant program via InforMutation exemplifies the improbability of evolution's modifying walking software to make it flying software.

InforMutation Versus Dawkins's WEASEL Program

Dawkins claimed that a computer program using a random mutation generator could in a feasible time convert a string of unordered letters into the phrase "METHINKS IT IS LIKE A WEASEL." Dawkins claimed this example showed amino acids could be ordered and reordered by random action in nature to produce complex proteins (Dawkins, 1986, pp. 47–48). (Readers may use a simulation of Dawkins's hypothetical program on the Internet; Maxwell, 2001).

Dawkins' program purports to show how random mutation and natural selection can produce a string of characters that carry meaning. His program operates by randomly selecting characters from an input string and arranging the output string in an attempt to match a predetermined pattern (e.g., the "WEASEL" sentence). Erroneous or nonmatching sequences are immediately discarded.

Differing considerably from Dawkins's WEASEL program, InforMuta-

Table I. Most common error messages and output results that appear when mutations are applied to the InforMutation simulation program.

Error Message or Output Result	Meaning or Significance
"Syntax Error"	The mutated program language instruction cannot be decoded and executed; part of the program may still run
"Unknown command"	The mutated program language instruction cannot be decoded or executed; part of the program may still run
"Line number not found"	The program instructions can no longer be executed to complete the operational cycle
"Divide by zero"	The program instruction attempted an operation that was impossible to complete
"Incomplete statement"	The mutated program instruction is malformed and cannot be decoded or executed
(outputs a series of numbers that are not all prime)	The mutated program instruction(s) no longer produce the original series; the user can judge whether the new result is "interesting" or an "improvement"
(output of numbers or text that are not the prime series)	The mutated program instruction(s) no longer produce the original series; the user can judge whether the new result is "interesting" or an "improvement"
(program ends without output)	The mutated program executes to some degree but provides no resulting information
(program operates without change in expected output)	The mutation did not affect a portion of the program that changes the result

tion does not operate on an input string and deliver an output string to fit a given pattern. Rather, InforMutation operates by mutating the software program itself and then testing the mutant program's capability or fitness to do anything at all. Unlike the WEASEL program, InforMutation does not weed out nonfunctioning mutations before testing the mutant program. In other words, unlike the WEASEL program, there is the potential for failure.

Dawkins claimed success for his program when its output result matched the predetermined "WEASEL" sentence. Dawkins's program never changes during operation. In contrast, the prime number program in InforMutation starts out functioning properly and then undergoes mutations that might make it perform other, different tasks.

Can InforMutation successfully mutate the prime number program to produce something else? Random software program mutations, as occur during InforMutation's simulation, deliver the types of results listed in Table II below. A manual tally of such results can provide an estimate of how often mutant programs appear that properly function and do something different from the original. To date, the results of InforMutation simulations show that only about 5% of mutant programs can even operate. Furthermore, no mutant program has done anything other than some variation, sometimes truncated, of counting from 1 to 101. Programs mutated more than once have nearly always been nonfunctional.

Table II. The fate of computer programs.

Action Imposed on a Properly Working Program	What the Program Does as a Result
Nothing	Operates properly and gives correct results
Make a change	Does not operate; halts with an error code
Make a change	Operates but gives the wrong results
Make a change	Gives the correct results and uses fewer resources or runs faster
Make a change	Gives the correct results but uses more resources or runs slower
Make a change	Works properly with no observable change in operation or results
Make a change	Produces different, desirable results
Make a change	(The change having reversed out or corrected a previous change) the program returns to proper operation and correct results

Summary and Conclusion

Even if new physical features, such as organs, limbs, and eyes, could arise via random mutation and natural selection, the knowledge and information necessary to use the new physical feature must arise simultaneously. For natural selection to favor a new or changed physical feature, the organism must be able to use the new feature in a beneficial manner. A new or transitional structure that is useless would likely interfere with survival, so natural selection would delete it (Bergman, 2005). A reptile with fully formed wings (hardware), for example, lacks a survival advantage compared to its wingless brethren unless it has the knowledge (software) to operate its wings in a way that improves its survival and enhances its ability to reproduce and pass the wing-trait to its offspring.

Biological "how-to" knowledge is directly analogous to computer software. In both biological and computer environments, carrying out a function is a step-by-step process that is coded and then executed (Behe, 1996, p. 41; Berlinski, 2000, pp. xii, xvi, 313–315; Freeland and Hurst, 2004). When applied to computer software programs, random mutations nearly always hobble or destroy a program's original function. The InforMutation simulation system allows direct observation of the nonconstructive results of such mutations on even simple programs. Experience with InforMutation has shown most software mutations damage the software so that the mutated program either does not deliver the correct (or "improved") functional results, or does not run at all.

If random mutation can only harm the function of software, then random mutation lacks the creative power to install or modify the biological software to operate legs, wings, or any other newly evolved physical feature. If random mutation cannot create the software for a new physical feature, then natural selection cannot favor that feature in the evolutionary process. Finally, if evolution theory cannot explain the creation of control software needed to use new physical features, then the theory cannot sustain its claim that undirected mutation and selection processes gave rise to the diversity of all physical features in all life on earth.

Acknowledgments

I deeply appreciate the careful review and feedback on later drafts from Jerry Bergman and several *CRSQ* reviewers, and on earlier drafts from Joseph Kezele and Roger Heyen of the Arizona Origin Science Association and from Kenneth Poppe. For quickly, accurately, and efficiently programming and implementing the InforMutation simulation system on PC and Internet, many thanks to Stephen E. Stern of Titus Information Systems.

References

Behe, M. 1996. *Darwin's Black Box*. Simon & Schuster, New York, NY.

Bergman, J. 2001. Why Dawkins' weasel demonstrates mutations cannot produce a new functional gene. *Creation Ex Nihilo Technical Journal* 15(2):69–76.

Bergman, J. 2005. Can evolution produce new organs or structures? *Creation Ex Nihilo Technical Journal* 19(2):76–82.

Berlinski, D. 2000. *The Advent of the Algorithm*. Harcourt, New York, NY.

Carroll, S.B. 2006. *The Making of the Fittest: DNA and the Ultimate Forensic Record of Evolution*. W. W. Norton & Co., New York, NY.

Clark, J.M. 2007. Dinosaurs and the origin of birds. In *Encyclopedia of Life Sciences*, 22:307–313. John Wiley & Sons, Hoboken, NJ .

Damuth, J.D. 2001. Evolution: tempo and mode. In *Encyclopedia of Life Sciences*, 6:705–711. Nature Publishing Group, London, UK.

Dawkins, R. 1986. *The Blind Watchmaker*. W.W. Norton, New York, NY.

Dawkins, R. 1995. *River Out of Eden: A Darwinian View of Life*. Basic Books, New York, NY.

Dembski, W.A. 2002. *No Free Lunch*. Rowan & Littlefield Publishers, Lanham, MD.

Devlin, K. 2005. *The Math Instinct*. Thunder's Mouth Press, New York, NY.

Dial, K.P., J.R. Randall, and T.R. Dial. 2006. What use is half a wing in the ecology and evolution of birds? *BioScience* 56:437–445.

Freeland, S.J., and L.D. Hurst. 2004. Evolution encoded. *Scientific American* 290:4:84–91.

Gould, S.J. 1996. *Full House*. Harmony Books, New York, NY.

Greenspan, R.J. 2002. Genetics of behavior. In Baltimore, D. (editor), *Frontiers of Life*, 1:595–596. Academic Press, San Diego, CA.

Hayden, T. 1999. The really early birds. *Newsweek 133(20):80.*

Hecht, J. 2000. Feathers fly. New Scientist 167(2245):18.

Kutschera, U., and K.J. Niklas. 2004. The modern theory of biological evolution: an expanded synthesis. *Naturwissenscchaften* 91:255–276. (http://www.evolutionsbiologen.de/niklas.pdf)

Lenski, R.E., C. Ofria, R.T. Pennock, and C. Adami. 2003. The evolutionary origins of complex features. *Nature* 423:139–144.

Levitin, D.J. 2006. *This Is Your Brain on Music: The Science of a Human Obsession*. Dutton, New York, NY.

Maxwell, S. 2001. *The Weasel Applet*. Online simulation at http://home.pacbell.net/s-max/scott/weasel.html (as of December 2, 2008).

Mlot, C. 1997. Evolutionary origins of fish antifreeze. *Science News* 151(16):237.

Padian, K. 2001. Cross-testing adaptive hypotheses: phylogenetic analysis and the origin of bird flight. *American Zoologist* 41:598–607.

Pinker, S. 1994. *The Language Instinct*. William Morrow & Co., New York, NY.

Pinker, S. 1997. *How the Mind Works*. W. W. Norton, New York, NY.

Poppe, K. 2005. *Reclaiming Science from Darwinism*. Harvest House, Eugene, OR.

Ridley, M. 2002. Natural selection: an overview. In Pagel, M. (editor), *Encyclopedia of Evolution*, 2:798. Oxford University Press, New York, NY.

Sagan, C. 1997. *Contact*. Pocket Publishers, New York, NY.

Sniegowski, P.D. 2002. Mutation: an overview. In Pagel, M. (editor), *Encyclopedia of Evolution*, 2:780. Oxford University Press, New York, NY.

Southwood, T.R.E. 2003. *The Story of Life*. Oxford University Press, New York, NY.

Spetner, L. 1999. *Not By Chance!* Judaica Press, New York, NY.

Stevens, R.W. 2007. *InforMutation*. On-line simulation at http://www.informutation.com (as of December 2, 2008).

Van Biema, D. 2006. God vs. science (interview of Richard Dawkins). *Time* 168:49–55.

APPENDIX I: SUMMARY OF ARGUMENT

The argument presented in this paper can be summarized as follows.

A. Causation Principle—Hardware—Software

1. Every operation of equipment ("hardware"), whether it be computer or biological, follows a series of cause-effect steps.
2. To direct the operation of any hardware, whether computer or biological, requires a stored set of instructions to direct the cause-effect steps. The instructions must be retrieved, decoded, and executed. The set of instructions is "software."
3. Considering the example of reptile-bird evolution: to operate a wing requires a different set of cause-effect steps from those needed to operate a leg.
4. To operate a wing therefore requires software different from that required to operate a leg.

B. Evolution requires random mutation to software to operate mutated hardware

5. Evolution posits random mutation as the agent causing changes to biological hardware.
6. To operate changed hardware requires changed software.
7. To change software requires chang-

ing elements of the stored set of instructions.

8. Evolution states all changes to biological elements are random (undirected).
9. For evolution to change the software to support the changed hardware—in a manner necessary for the changed hardware to confer a survival advantage—there must be applied random (undirected) changes to the stored set of instructions.

C. Computer software is sufficiently analogous to biological software to show mutations to such software cannot produce needed changes to support new hardware

10. A computer program is a stored set of instructions.
11. Evolutionists affirm: (a) DNA is a software system containing stored instructions; (b) the operation of DNA is computer-like, albeit quaternary rather than binary; (c) the brain operates in a manner that bears comparison to a modern digital computer; and (d) computer models of evolutionary mutation and natural selection afford a valid comparison to biological evolution.
12. Applying random mutation to the stored instructions in a computer program is therefore acceptably analogous to applying random mutation to the stored instructions in DNA and other biological structures that store instructions for later retrieval, e.g., the brain and nervous system.
13. The InforMutation system shows what happens when you apply random (undirected) mutations to a set of stored instructions.
14. Nearly always, random mutations applied to a stored set of instructions (analogous to biological stored instructions) damages the instructions, by causing the instructions to either: (a) not decode properly; (b) not execute fully or at all; or (c) not execute in a manner that produces a useful, beneficial, creative, or "interesting" change in the operation of the hardware.

D. If software cannot be modified by random mutation to support new features and hardware under conditions of natural selection, then the change agent underlying evolution is too weak to support its claims

15. If computer software cannot be randomly mutated successfully to cause it to do anything qualitatively different from its current operation, then it cannot be randomly mutated to operate different hardware or accomplish some other qualitatively different task.
16. By analogy, if computer software cannot be randomly mutated successfully, then there is no evident reason to believe biological software can be randomly mutated successfully to operate new or qualitatively different hardware.
17. Therefore, there is no reason to believe random mutation can modify the reptile's leg software to operate wings at the very time that such software is needed to confer the wing's survival advantage.
18. If a new hardware component, e.g., the wing, cannot be supported immediately by software and thus confers no survival advantage, then the hardware evolution of that new component will likely cease; and by necessity, so will any evolution of the software also cease.
19. Given the twin improbabilities of hardware evolution and coincident operating software evolution, the evolutionary method of producing new biological features (and thus new species) loses all plausibility as the creative change agent producing all species.