Ausgewählte Methoden der Künstlichen Intelligenz

Vorlesung 9, Praktikum

Prof. Dr. Tatyana Ivanovska

<t.ivanovska@oth-aw.de>

9. Januar 2024

Ostbayerische Technische Hochschule
Amberg-Weiden

Thema für heute

- Weitere Simulationen mit Agenten
- Die Evolutionstheorie

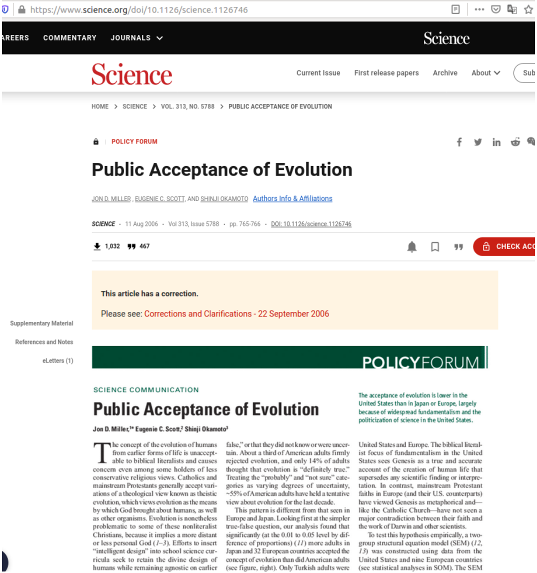
Die Evolutionstheorie

- Die wichtigste Idee in der Biologie ist die Theorie der Evolution durch natürliche Selektion
- Neue Arten entstehen und vorhandene Arten verändern sich durch natürliche Selektion

Glauben die Menschen an die Evolution?

Was denken Sie?

Die öffentliche Akzeptanz der Evolution



https://www.science.org/doi/10.1126/science.1126746

AREERS COMMENTARY JOURNALS

Science

Current Issue First release papers Archive About Sub

HOME > SCIENCE > VOL. 313, NO. 5788 > PUBLIC ACCEPTANCE OF EVOLUTION

POLICY FORUM

Public Acceptance of Evolution

JON D. MILLER, EUGENIE C. SCOTT, AND SHINJI OKAMOTO Authors Info & Affiliations

SCIENCE • 11 Aug 2006 • Vol 313, Issue 5788 • pp. 765-766 • DOI: 10.1126/science.1126746

This article has a correction.

Please see: Corrections and Clarifications - 22 September 2006

Supplementary Material

References and Notes

eLetters (1)

POLICYFORUM

SCIENCE COMMUNICATION

Public Acceptance of Evolution

Jon D. Miller,[1*] Eugenie C. Scott,[2] Shinji Okamoto[3]

The acceptance of evolution is lower in the United States than in Japan or Europe, largely because of widespread fundamentalism and the politicization of science in the United States.

The concept of the evolution of humans from earlier forms of life is unacceptable to biblical literalists and causes concern even among some holders of less conservative religious views. Catholics and mainstream Protestants generally accept variations of a theological view known as theistic evolution, which views evolution as the means by which God brought about humans, as well as other organisms. Evolution is nonetheless problematic to some of these nonliteralist Christians, because it implies a more distant or less personal God (*1–3*). Efforts to insert "intelligent design" into school science curricula seek to retain the divine design of humans while remaining agnostic on earlier false," or that they did not know or were uncertain. About a third of American adults firmly rejected evolution, and only 14% of adults thought that evolution is "definitely true." Treating the "probably" and "not sure" categories as varying degrees of uncertainty, ~55% of American adults have held a tentative view about evolution for the last decade.

This pattern is different from that seen in Europe and Japan. Looking first at the simpler true-false question, our analysis found that significantly (at the 0.01 to 0.05 level by difference of proportions) (*11*) more adults in Japan and 32 European countries accepted the concept of evolution than did American adults (see figure, right). Only Turkish adults were United States and Europe. The biblical literalist focus of fundamentalism in the United States sees Genesis as a true and accurate account of the creation of human life that supersedes any scientific finding or interpretation. In contrast, mainstream Protestant faiths in Europe (and their U.S. counterparts) have viewed Genesis as metaphorical and—like the Catholic Church—have not seen a major contradiction between their faith and the work of Darwin and other scientists.

To test this hypothesis empirically, a two-group structural equation model (SEM) (*12, 13*) was constructed using data from the United States and nine European countries (see statistical analyses in SOM). The SEM

Die öffentliche Akzeptanz der Evolution

Ostbayerische Technische Hochschule Amberg-Weiden

POLICYFORUM

SCIENCE COMMUNICATION

Public Acceptance of Evolution

The acceptance of evolution is lower in the United States than in Japan or Europe, largely because of widespread fundamentalism and the politicization of science in the United States.

Jon D. Miller,[1*] Eugenie C. Scott,[2] Shinji Okamoto[3]

The concept of the evolution of humans from earlier forms of life is unacceptable to biblical literalists and causes concern even among some holders of less conservative religious views. Catholics and mainstream Protestants generally accept variations of a theological view known as theistic evolution, which views evolution as the means by which God brought about humans, as well as other organisms. Evolution is nonetheless problematic to some of these nonliteralist Christians, because it implies a more distant or less personal God (*1–3*). Efforts to insert "intelligent design" into school science curricula seek to retain the divine design of humans while remaining agnostic on earlier creationist beliefs in a young Earth and the coexistence of humans and dinosaurs (*2*, *4*).

Beginning in 1985, national samples of U.S. adults have been asked whether the statement, "Human beings, as we know them, developed from earlier species of animals," is true or false, or whether the respondent is not sure or does not know. We compared the results of these surveys with survey data from nine European countries in 2002, surveys in 32 European countries in 2005, and a national survey in Japan in 2001 (*5*). Over the past 20 years, the percentage of U.S. adults accepting the idea of evolution has declined from 45% to 40% and the percentage of adults overtly rejecting evolution declined from 48% to 39%. The percentage of adults who were not sure about evolution increased from 7% in 1985 to 21% in 2005. After 20 years of public debate, the public appears to be divided evenly in terms of accepting or rejecting evolution, with about one in five adults still undecided or unaware of the issue. This pattern is consistent with a number of sporadic national newspaper surveys reported in recent years (*6–10*).

A dichotomous true-false question format tends to exaggerate the strength of both positions. In 1993 and 2003, national samples of American adults were asked about the same statement but were offered the choice of saying that the statement was "definitely true, probably true, probably false, definitely false," or that they did not know or were uncertain. About a third of American adults firmly rejected evolution, and only 14% of adults thought that evolution is "definitely true." Treating the "probably" and "not sure" categories as varying degrees of uncertainty, ~55% of American adults have held a tentative view about evolution for the last decade.

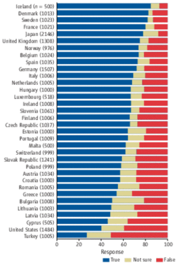
This pattern is different from that seen in Europe and Japan. Looking first at the simpler true-false question, our analysis found that significantly (at the 0.01 to 0.05 level by difference of proportions) (*11*) more adults in Japan and 32 European countries accepted the concept of evolution than did American adults (see figure, right). Only Turkish adults were less likely to accept the concept of evolution than American adults. In Iceland, Denmark, Sweden, and France, 80% or more of adults accepted the concept of evolution, as did 78% of Japanese adults.

A cross-national study of the United States and nine European nations in 2002–2003 used the expanded version of the question. The results confirm that a significantly lower proportion of American adults believe that evolution is absolutely true than adults in nine European countries [see fig. S1 in the Supporting Online Material (SOM)]. A third of American adults indicated that evolution is "absolutely false"; the proportion of European adults who thought that evolution was absolutely false ranged from 7% in Denmark, France, and Great Britain to 15% in the Netherlands.

Regardless of the form of the question, one in three American adults firmly rejects the concept of evolution, a significantly higher proportion than found in any western European country. How can we account for this pattern of American reservations about the concept of evolution in the context of broad acceptance in Europe and Japan?

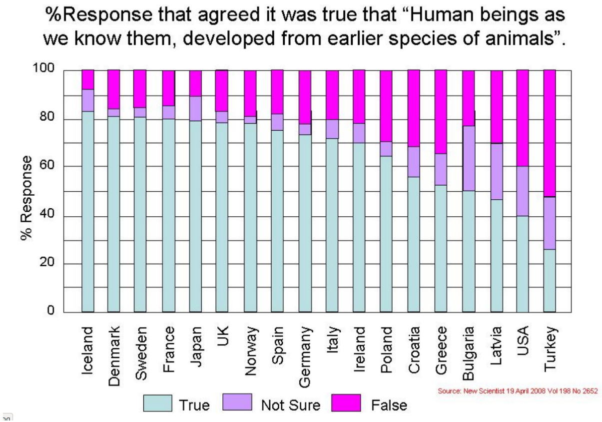
First, the structure and beliefs of American fundamentalism historically differ from those of mainstream Protestantism in both the United States and Europe. The biblical literalist focus of fundamentalism in the United States sees Genesis as a true and accurate account of the creation of human life that supersedes any scientific finding or interpretation. In contrast, mainstream Protestant faiths in Europe (and their U.S. counterparts) have viewed Genesis as metaphorical and—like the Catholic Church—have not seen a major contradiction between their faith and the work of Darwin and other scientists.

To test this hypothesis empirically, a two-group structural equation model (SEM) (*12*, *13*) was constructed using data from the United States and nine European countries (see statistical analyses in SOM). The SEM allows an examination of the relation between several variables simultaneously on one or more outcome variables. In this model, 10 independent variables—age, gender, education, genetic literacy, religious belief, attitude toward life, attitude toward science and tech-



[1]Michigan State University, East Lansing, MI 48824–1115, USA. [2]National Center for Science Education, Oakland, CA 94609, USA. [3]Kobe University, Rokkaido, Hyogo, Japan.

Die öffentliche Akzeptanz der Evolution



Wir werden heute die Evolution simulieren

- Ein einfaches Modell, das eine Grundform der Evolution demonstriert
- Replikatoren
  - Eine Population von Agenten, die sich fortpflanzen können
- Variation (Veränderung)
  - Unterschiede zwischen Individuen in einer Population
- Unterschiedliches Überleben oder Reproduktion
  - Die Unterschiede zwischen den Individuen müssen sich auf ihre Fähigkeit zum Überleben und zur Fortpflanzung auswirken.

Modell

- Eine Population von Agenten, die einzelne Organismen darstellen
- Jeder Agent hat eine genetische Information (den Genotyp).
- In unserem Modell wird ein Genotyp durch eine Folge von N binären Ziffern (0 und 1) dargestellt, N ist ein Hyperparameter
- Eine Population hat eine Vielzahl von Genotypen
- Um ein unterschiedliches Überleben zu erzeugen, definieren wir eine Funktion, die jeden Genotyp auf eine Fitness abbildet, wobei Fitness eine Größe ist, die sich auf die Fähigkeit eines Agenten bezieht, zu überleben oder sich zu vermehren

Fitness-Landschaft

- In der Evolutionsbiologie werden Fitnesslandschaften verwendet, um die Beziehung zwischen Genotypen und Fortpflanzungserfolg zu veranschaulichen. (`https://en.wikipedia.org/wiki/Fitness_landscape`)
- Die Idee, die Evolution zu untersuchen, indem man die Verteilung der Fitnesswerte als eine Art Landschaft visualisiert, wurde erstmals von Sewall Wright (ein amerikanischer Genetiker, der für seine einflussreichen Arbeiten zur Evolutionstheorie bekannt ist) im Jahr 1932 vorgestellt

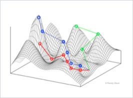
Genotype to fitness landscapes [edit]

Wright visualized a genotype space as a hypercube.[1] No continuous genotype "dimension" is defined. Instead, a network of genotypes are connected via mutational paths.

Stuart Kauffman's NK model falls into this category of fitness landscape. Newer network analysis techniques such as selection-weighted attraction graphing (SWAG) also use a dimensionless genotype space.[5]

Allele frequency to fitness landscapes [edit]

Wright's mathematical work described fitness as a function of allele frequencies.[2] Here, each dimension describes an allele frequency at a different gene, and goes between 0 and 1.



Visualization of two dimensions of an NK fitness landscape. The arrows represent various mutational paths that the population could follow while evolving on the fitness landscape.

Fitness-Landschaft

- In der realen Welt sind die Fitnesslandschaften kompliziert
- Für unsere einfache Simulation benötigen wir eine Beziehung zwischen Genotyp und Fitness.
- Es stellt sich heraus, dass es jede beliebige Beziehung sein kann
- Um dies zu demonstrieren, werden wir eine völlig zufällige Fitnesslandschaft verwenden

Fitness-Landschaft

```python
class FitnessLandscape:

    def __init__(self, N):
        """Create a fitness landscape.

        N: number of dimensions
        """
        self.N = N
        self.set_values()

    def set_values(self):
        self.one_values = np.random.random(self.N)
        self.zero_values = np.random.random(self.N)

    def random_loc(self):
        """Choose a random location."""
        return np.random.randint(2, size=self.N, dtype=np.int8)

    def fitness(self, loc):
        """Evaluates the fitness of a location.

        loc: array of N 0s and 1s

        returns: float fitness
        """
        fs = np.where(loc, self.one_values, self.zero_values)
        return fs.mean()

    def distance(self, loc1, loc2):
        return np.sum(np.logical_xor(loc1, loc2))
```

Quelle: A. Downey, Think Complexity

Agent

```python
class Agent:
    """Represents an agent in an NK model."""

    def __init__(self, loc, fit_land):
        """Create an agent at the given location.

        loc: array of N 0s and 1s
        fit_land: reference to an fit_land
        """
        self.loc = loc
        self.fit_land = fit_land
        self.fitness = fit_land.fitness(self.loc)

    def copy(self):
        return Agent(self.loc, self.fit_land)
```

Each agent has a location, a reference to a FitnessLandscape, and a fitness.

Quelle: A. Downey, Think Complexity

Simulation

The Simulator

The `Simulator` class provides methods to run the simulations.

```
class Simulation:

    def __init__(self, fit_land, agents):
        """Create the simulation:

        fit_land: fit_land
        num_agents: int number of agents
        agent_maker: function that makes agents
        """
        self.fit_land = fit_land
        self.agents = np.asarray(agents)
        self.instruments = []

    def add_instrument(self, instrument):
        """Adds an instrument to the list.

        instrument: Instrument object
        """
        self.instruments.append(instrument)

    def plot(self, index, *args, **kwargs):
        """Plot the results from the indicated instrument.
        """
        self.instruments[index].plot(*args, **kwargs)

    def run(self, num_steps=500):
        """Run the given number of steps.

        num_steps: integer
        """
        # initialize any instruments before starting
        self.update_instruments()

        for _ in range(num_steps):
            self.step()
```

Quelle: A. Downey, Think Complexity

Simulation

```python
def step(self):
    """Simulate a time step and update the instruments.
    """
    n = len(self.agents)
    fits = self.get_fitnesses()

    # see who dies
    index_dead = self.choose_dead(fits)
    num_dead = len(index_dead)

    # replace the dead with copies of the living
    replacements = self.choose_replacements(num_dead, fits)
    self.agents[index_dead] = replacements

    # update any instruments
    self.update_instruments()

def update_instruments(self):
    for instrument in self.instruments:
        instrument.update(self)

def get_locs(self):
    """Returns a list of agent locations."""
    return [tuple(agent.loc) for agent in self.agents]

def get_fitnesses(self):
    """Returns an array of agent fitnesses."""
    fits = [agent.fitness for agent in self.agents]
    return np.array(fits)

def choose_dead(self, ps):
    """Choose which agents die in the next timestep.

    ps: probability of survival for each agent

    returns: indices of the chosen ones
    """
    n = len(self.agents)
    is_dead = np.random.random(n) < 0.1
    index_dead = np.nonzero(is_dead)[0]
    return index_dead
```

Quelle: A. Downey, Think Complexity

Simulation

```python
def choose_replacements(self, n, weights):
    """Choose which agents reproduce in the next timestep.

    n: number of choices
    weights: array of weights

    returns: sequence of Agent objects
    """
    agents = np.random.choice(self.agents, size=n, replace=True)
    replacements = [agent.copy() for agent in agents]
    return replacements
```

Quelle: A. Downey, Think Complexity

Agentenerstellung

We'll use a few functions to create agents. The first one starts with identical agents:

```
def make_identical_agents(fit_land, num_agents, agent_maker):
    """Make an array of Agents.

    fit_land: FitnessLandscape
    num_agents: integer
    agent_maker: class used to make Agent

    returns: array of Agents
    """
    loc = fit_land.random_loc()
    agents = [agent_maker(loc, fit_land) for _ in range(num_agents)]
    return np.array(agents)
```

Here's another that starts with agents at random locations:

```
def make_random_agents(fit_land, num_agents, agent_maker):
    """Make an array of Agents.

    fit_land: FitnessLandscape
    num_agents: integer
    agent_maker: class used to make Agent

    returns: array of Agents
    """
    locs = [fit_land.random_loc() for _ in range(num_agents)]
    agents = [agent_maker(loc, fit_land) for loc in locs]
    return np.array(agents)
```

Quelle: A. Downey, Think Complexity

Agentenerstellung

And this one puts one agent at every possible location:

```python
import itertools

def make_all_agents(fit_land, agent_maker):
    """Make an array of Agents.

    fit_land: FitnessLandscape
    agent_maker: class used to make Agent

    returns: array of Agents
    """
    N = fit_land.N
    locations = itertools.product([0, 1], repeat=N)
    agents = [agent_maker(loc, fit_land) for loc in locations]
    return np.array(agents)
```

Quelle: A. Downey, Think Complexity

MeanFitness als Metric

Instruments

To measure these changes over the course of the simulations, we'll use Instrument objects.

```python
class Instrument:
    """Computes a metric at each timestep."""

    def __init__(self):
        self.metrics = []

    def update(self, sim):
        """Compute the current metric.

        Appends to self.metrics.

        sim: Simulation object
        """
        # child classes should implement this method
        pass

    def plot(self, **options):
        plt.plot(self.metrics, **options)
```

The `MeanFitness` instrument computes the mean fitness after each time step.

```python
class MeanFitness(Instrument):
    """Computes mean fitness at each timestep."""
    label = 'Mean fitness'

    def update(self, sim):
        mean = np.nanmean(sim.get_fitnesses())
        self.metrics.append(mean)
```

Quelle: A. Downey, Think Complexity

Jetzt kann man das einfachste Model starten

Ohne unterschiedliches Überleben oder Fortpflanzung ergibt sich eine Zufallsbewegung.

- Bitte laufen Sie den Quellcode und plotten die MeanFitness-Werte

Differential Survival

We can add differential survival by overriding `choose_dead`

```
5]
class SimWithDiffSurvival(Simulation):

    def choose_dead(self, ps):
        """Choose which agents die in the next timestep.

        ps: probability of survival for each agent

        returns: indices of the chosen ones
        """
        n = len(self.agents)
        is_dead = np.random.random(n) > ps
        index_dead = np.nonzero(is_dead)[0]
        return index_dead
```

Quelle: A. Downey, Think Complexity

Differential Survival

```python
class SimWithDiffReproduction(Simulation):

    def choose_replacements(self, n, weights):
        """Choose which agents reproduce in the next timestep.

        n: number of choices
        weights: array of weights

        returns: sequence of Agent objects
        """
        p = weights / np.sum(weights)
        agents = np.random.choice(self.agents, size=n, replace=True, p=p)
        replacements = [agent.copy() for agent in agents]
        return replacements
```

Quelle: A. Downey, Think Complexity

Mutation

Mutation is one way of increasing, or at least maintaining, diversity.

`Mutant` is a kind of agent that overrides `copy`:

```
class Mutant(Agent):

    def copy(self, prob_mutate=0.05):
        if np.random.random() > prob_mutate:
            loc = self.loc.copy()
        else:
            direction = np.random.randint(self.fit_land.N)
            loc = self.mutate(direction)
        return Mutant(loc, self.fit_land)

    def mutate(self, direction):
        """Computes the location in the given direction.

        Result differs from the current location along the given axis.

        direction: int index from 0 to N-1

        returns: new array of N 0s and 1s
        """
        new_loc = self.loc.copy()
        new_loc[direction] ^= 1
        return new_loc
```

Quelle: A. Downey, Think Complexity

Aufgaben

- Starten Sie N Simulationen mit unt. Anzahl von Agenten und plotten die Metrik pro Timestep
- Wie ändert sich das Mean-Fitness?
- Wie wichtig sind die Hyperparameter, z.B. Mutation-Rate?
- Ändert sich etwas, wenn man mit den gleichen Agenten startet?

Danke für die Aufmerksamkeit!