Lecture 2: Numbers, Errors, Chaos

David Lester

2017

Outline

Introduction

At the end of this lecture you will:

- Be aware of different sorts of mathematical arithemtic systems.

Introduction

At the end of this lecture you will:

- Be aware of different sorts of mathematical arithemtic systems.
- Have a feeling for countability arguments.

Introduction

At the end of this lecture you will:

- Be aware of different sorts of mathematical arithemtic systems.
- Have a feeling for countability arguments.
- Be aware of the practical problems that occur when using the usual numbers provided on a computer.

Introduction

At the end of this lecture you will:

- Be aware of different sorts of mathematical arithemtic systems.
- Have a feeling for countability arguments.
- Be aware of the practical problems that occur when using the usual numbers provided on a computer.
- Be aware of the issues and limitations arising when using numeric simulations.

Introduction

At the end of this lecture you will:

- Be aware of different sorts of mathematical arithemtic systems.
- Have a feeling for countability arguments.
- Be aware of the practical problems that occur when using the usual numbers provided on a computer.
- Be aware of the issues and limitations arising when using numeric simulations.
- Be alert to the issue of chaotic behaviour, which we might expect to be common in brain simulation.

Natural Numbers ($\mathbb{N}$)

Die ganzen Zahlen hat der liebe Gott gemacht,
alles andere ist Menschenwerk.

Leopold Kronecker (1823-1891)

- These are the simplest kind of number: the counting numbers $\{1,\ 2,\ 3,\ \ldots\}$.

Natural Numbers ($\mathbb{N}$)

Die ganzen Zahlen hat der liebe Gott gemacht,
alles andere ist Menschenwerk.

Leopold Kronecker (1823-1891)

- These are the simplest kind of number: the counting numbers $\{1, 2, 3, \ldots\}$.
- Computer Scientists (who start at 0) and Mathematicians (who usually start at 1) differ on where to start!

Natural Numbers ($\mathbb{N}$)

Die ganzen Zahlen hat der liebe Gott gemacht,
alles andere ist Menschenwerk.
Leopold Kronecker (1823-1891)

- These are the simplest kind of number: the counting numbers $\{1,\ 2,\ 3,\ \ldots\}$.
- Computer Scientists (who start at 0) and Mathematicians (who usually start at 1) differ on where to start!
- One *implementation* of the natural numbers is *Peano Arithmetic*.

Peano Arithmetic

Definition

A Peano number is either Z (for zero), or if n is a Peano number, then $S\ n$ (for successor) is also a Peano Number.
Together Z and S define a recursive data type.

Example

We can implement addition with the following recursive function (add).

$$\begin{aligned} add(Z, \quad m) &= m \\ add(S\ n, \quad m) &= add(n, S\ m) \end{aligned}$$

Peano Arithmetic

Exercise

Show how to implement multiplication. **Hint** use the definition of *add* above.

Exercise

Show how to implement the power function n^m. **Hint** use the definition of *mul* above.

How many counting numbers are there?

Theorem

There are infinitely many counting numbers.

Proof

- Assume that there is only a finite number of elements of $\mathbb{N}$.

$\square$

How many counting numbers are there?

Theorem

There are infinitely many counting numbers.

Proof

- Assume that there is only a finite number of elements of $\mathbb{N}$.
- Then there must be a biggest one. Call it n.

□

How many counting numbers are there?

Theorem

There are infinitely many counting numbers.

Proof

- Assume that there is only a finite number of elements of $\mathbb{N}$.
- Then there must be a biggest one. Call it n.
- But by Peano Arithmetic, $S\ n \in \mathbb{N}$, and $S\ n > n$.

□

How many counting numbers are there?

Theorem

There are infinitely many counting numbers.

Proof

- Assume that there is only a finite number of elements of $\mathbb{N}$.
- Then there must be a biggest one. Call it n.
- But by Peano Arithmetic, $S\ n \in \mathbb{N}$, and $S\ n > n$.
- So n is *not* the biggest element of $\mathbb{N}$ which is a contradiction.

$\square$

A Digression on Bijections

- Bijections are a very special kind of function from one set S (for source) to another set T (for target). We normally write $f : S \to T$ to remind ourselves which sets a function is mapping between. When I remember, I try to use the symbol ϕ for bijections.

A Digression on Bijections

- Bijections are a very special kind of function from one set S (for source) to another set T (for target). We normally write $f : S \to T$ to remind ourselves which sets a function is mapping between. When I remember, I try to use the symbol ϕ for bijections.
- The key property of bijections is that they map one element in the source set ($s \in S$) to one element in the target set ($t \in T$), and *vice versa*.

A Digression on Bijections

- Bijections are a very special kind of function from one set S (for source) to another set T (for target). We normally write $f : S \to T$ to remind ourselves which sets a function is mapping between. When I remember, I try to use the symbol ϕ for bijections.
- The key property of bijections is that they map one element in the source set ($s \in S$) to one element in the target set ($t \in T$), and *vice versa*.
- Because of this 'one-to-one' property, for every bijection $\phi : S \to T$ there is an inverse function $\phi^{-1} : T \to S$ with:

A Digression on Bijections

- Bijections are a very special kind of function from one set S (for source) to another set T (for target). We normally write $f : S \to T$ to remind ourselves which sets a function is mapping between. When I remember, I try to use the symbol ϕ for bijections.
- The key property of bijections is that they map one element in the source set ($s \in S$) to one element in the target set ($t \in T$), and *vice versa*.
- Because of this 'one-to-one' property, for every bijection $\phi : S \to T$ there is an inverse function $\phi^{-1} : T \to S$ with:
    - $\phi^{-1} : T \to S$ is itself a bijection;

A Digression on Bijections

- Bijections are a very special kind of function from one set S (for source) to another set T (for target). We normally write $f : S \to T$ to remind ourselves which sets a function is mapping between. When I remember, I try to use the symbol ϕ for bijections.
- The key property of bijections is that they map one element in the source set ($s \in S$) to one element in the target set ($t \in T$), and *vice versa*.
- Because of this 'one-to-one' property, for every bijection $\phi : S \to T$ there is an inverse function $\phi^{-1} : T \to S$ with:
  - $\phi^{-1} : T \to S$ is itself a bijection;
  - $\forall (s \in S) : (\phi^{-1} \circ \phi)(s) = s$; and

A Digression on Bijections

- Bijections are a very special kind of function from one set S (for source) to another set T (for target). We normally write $f : S \to T$ to remind ourselves which sets a function is mapping between. When I remember, I try to use the symbol ϕ for bijections.
- The key property of bijections is that they map one element in the source set ($s \in S$) to one element in the target set ($t \in T$), and *vice versa*.
- Because of this 'one-to-one' property, for every bijection $\phi : S \to T$ there is an inverse function $\phi^{-1} : T \to S$ with:
  - $\phi^{-1} : T \to S$ is itself a bijection;
  - $\forall(s \in S) : (\phi^{-1} \circ \phi)(s) = s$; and
  - $\forall(t \in T) : (\phi \circ \phi^{-1})(t) = t$;

An Important Bijection Property

Definition

If there is a bijection between two sets A and B, then these two sets have the same *cardinality*. We write

$$\mathrm{Card}(A) = \mathrm{Card}(B)$$

For finite sets this simply says that the two sets A and B must have the same number of elements.

Integers ($\mathbb{Z}$)

- The set of integers extends the set of natural numbers by including negative number of the form $-n$ where $n \in \mathbb{N}$.

Integers ($\mathbb{Z}$)

- The set of integers extends the set of natural numbers by including negative number of the form $-n$ where $n \in \mathbb{N}$.
- The *Algebraic Structure* $(\mathbb{Z}, 0, +)$ is an *Abelian Group*.

Integers ($\mathbb{Z}$)

- The set of integers extends the set of natural numbers by including negative number of the form $-n$ where $n \in \mathbb{N}$.
- The *Algebraic Structure* $(\mathbb{Z}, 0, +)$ is an *Abelian Group*.

  Identity: 0 is a left and right identity element for the $+$ operation.

Integers ($\mathbb{Z}$)

- The set of integers extends the set of natural numbers by including negative number of the form $-n$ where $n \in \mathbb{N}$.
- The *Algebraic Structure* $(\mathbb{Z}, 0, +)$ is an *Abelian Group*.

  Identity 0 is a left and right identity element for the $+$ operation.

  Associativity $(x + y) + z = x + (y + z)$.

Integers ($\mathbb{Z}$)

- The set of integers extends the set of natural numbers by including negative number of the form $-n$ where $n \in \mathbb{N}$.
- The *Algebraic Structure* $(\mathbb{Z}, 0, +)$ is an *Abelian Group*.

Identity	0 is a left and right identity element for the $+$ operation.
Associativity	$(x + y) + z = x + (y + z)$.
Inverse	For every $x \in \mathbb{Z}$, there is a $y \in \mathbb{Z}$, with $x + y = 0$. (Choose $y = -x$.)

Integers ($\mathbb{Z}$)

- The set of integers extends the set of natural numbers by including negative number of the form $-n$ where $n \in \mathbb{N}$.
- The *Algebraic Structure* $(\mathbb{Z}, 0, +)$ is an *Abelian Group*.

  Identity 0 is a left and right identity element for the $+$ operation.

  Associativity $(x + y) + z = x + (y + z)$.

  Inverse For every $x \in \mathbb{Z}$, there is a $y \in \mathbb{Z}$, with $x + y = 0$. (Choose $y = -x$.)

  Commutitivity $x + y = y + x$.

Countability, or: bijections with subsets of $\mathbb{N}$

Definition

We say that a set A is *countable*, if there is a subset of the natural numbers $B \subseteq \mathbb{N}$ and a bijection $\phi : A \rightarrow B$.
If B is infinite (*i.e.* $B = \mathbb{N}$), then we say that A is *countably infinite*. If there is no bijection ϕ then the set a is *uncountable*.

Enumerations

Corollary

If A is countably infinite, then there exists an enumeration *of the set A, i.e. A can be written out as:*

$$\{a_0,\ a_1,\ a_2,\ \dots\ , a_k,\ \dots\ \}$$

Proof

Let $a_n = \phi^{-1}(n)$.

□

How many integers are there?

Theorem

There are countably infinitely many integers.

Proof

Let $\phi : \mathbb{Z} \to \mathbb{N}$ be the function

$$\phi(i) = \begin{cases} 2i & \text{if } i \geq 0 \\ -2i - 1 & \text{if } i < 0 \end{cases}$$

In short: the negative integers are mapped to odd natural numbers; non-negative integers are mapped to even natural numbers.

□

Rationals ($\mathbb{Q}$)

- The set of rational number $\mathbb{Q}$ is an extension of the integers, in which we permit the representation of fractions.

Rationals ($\mathbb{Q}$)

- The set of rational number $\mathbb{Q}$ is an extension of the integers, in which we permit the representation of fractions.
- Formally, $q \in \mathbb{Q}$ if there exists $n \in \mathbb{Z}$ and $d \in \mathbb{N} - \{0\}$, such that $q = n/d$.

Rationals ($\mathbb{Q}$)

- The set of rational number $\mathbb{Q}$ is an extension of the integers, in which we permit the representation of fractions.
- Formally, $q \in \mathbb{Q}$ if there exists $n \in \mathbb{Z}$ and $d \in \mathbb{N} - \{0\}$, such that $q = n/d$.
- The *Algebraic Structure* $(\mathbb{Q}, 0, 1, +, \times)$ is a *Field*.

Rationals ($\mathbb{Q}$)

- The set of rational number $\mathbb{Q}$ is an extension of the integers, in which we permit the representation of fractions.
- Formally, $q \in \mathbb{Q}$ if there exists $n \in \mathbb{Z}$ and $d \in \mathbb{N} - \{0\}$, such that $q = n/d$.
- The *Algebraic Structure* $(\mathbb{Q}, 0, 1, +, \times)$ is a *Field*.
  Abelian Additive Group $(\mathbb{Q}, 0, +)$ is an Abelian Group.

Rationals ($\mathbb{Q}$)

- The set of rational number $\mathbb{Q}$ is an extension of the integers, in which we permit the representation of fractions.
- Formally, $q \in \mathbb{Q}$ if there exists $n \in \mathbb{Z}$ and $d \in \mathbb{N} - \{0\}$, such that $q = n/d$.
- The *Algebraic Structure* $(\mathbb{Q}, 0, 1, +, \times)$ is a *Field*.

  Abelian Additive Group $(\mathbb{Q}, 0, +)$ is an Abelian Group.

  Abelian Multiplicative Group $(\mathbb{Q} - \{0\}, 1, \times)$ is an Abelian Group.

Rationals ($\mathbb{Q}$)

- The set of rational number $\mathbb{Q}$ is an extension of the integers, in which we permit the representation of fractions.
- Formally, $q \in \mathbb{Q}$ if there exists $n \in \mathbb{Z}$ and $d \in \mathbb{N} - \{0\}$, such that $q = n/d$.
- The *Algebraic Structure* $(\mathbb{Q}, 0, 1, +, \times)$ is a *Field*.

  Abelian Additive Group $(\mathbb{Q}, 0, +)$ is an Abelian Group.

  Abelian Multiplicative Group $(\mathbb{Q} - \{0\}, 1, \times)$ is an Abelian Group.

  Distributivity $x \times (y + z) = x \times y + x \times z$.

A bijection for pairs of natural numbers

Theorem

Rather surprisingly, there is a bijection that maps pairs of natural numbers to a single natural number.

$$\phi_X : (n,\ m) = 2^n(2m+1) - 1$$

The rationals are countable

Corollary

The rationals $\mathbb{Q}$ are countable.

Proof

1. First map the numerator n to a natural number n' using the bijection we develped to show that the integers are countable.

□

The rationals are countable

Corollary

The rationals $\mathbb{Q}$ are countable.

Proof

1. First map the numerator n to a natural number n' using the bijection we develped to show that the integers are countable.
2. Then use ϕ_x to map n' and $d-1$ to a natural number m.

$$m = \phi_x(n', d-1)$$

□

On the matter of computability

- So far we have been very mathematical.

On the matter of computability

- So far we have been very mathematical.
- What are the computer science implications?

On the matter of computability

- So far we have been very mathematical.
- What are the computer science implications?
- **Good News!** All of the number systems we have discussed so far are available to you.

On the matter of computability

- So far we have been very mathematical.
- What are the computer science implications?
- **Good News!** All of the number systems we have discussed so far are available to you.
  - In C and C++, download `gmp` from the gnu website.

On the matter of computability

- So far we have been very mathematical.
- What are the computer science implications?
- **Good News!** All of the number systems we have discussed so far are available to you.
  - In C and C++, download `gmp` from the gnu website.
  - In python use the `gmpy2` package; this links to gmp.

On the matter of computability

- So far we have been very mathematical.
- What are the computer science implications?
- **Good News!** All of the number systems we have discussed so far are available to you.
  - In C and C++, download `gmp` from the gnu website.
  - In python use the `gmpy2` package; this links to gmp.
  - Be warned! These libraries are much slower than usual computer arithmetic, but *if* they deliver an answer then that answer is (probably) correct.

Reals ($\mathbb{R}$)

- As you might imagine by now, we are going to extend the rational numbers.

Reals ($\mathbb{R}$)

- As you might imagine by now, we are going to extend the rational numbers.
- This time we add the innocuous-seeming *Axiom of Real Completeness*.

Reals ($\mathbb{R}$)

- As you might imagine by now, we are going to extend the rational numbers.
- This time we add the innocuous-seeming *Axiom of Real Completeness*.
- If $X \in \mathbb{R}$ is *bounded above* by y, that is every element of the set $z \in X$ satisfies $z \leq y$, then there exists a *least upper bound* $x \in \mathbb{R}$.

$\sqrt{2}$ is irrational

Theorem

The square root of 2 is not a rational number: $\sqrt{2} \notin \mathbb{Q}$.

Proof

1. Suppose on the contrary that $\sqrt{2} \in \mathbb{Q}$.

$\square$

$\sqrt{2}$ is irrational

Theorem

The square root of 2 *is not a rational number:* $\sqrt{2} \notin \mathbb{Q}$.

Proof

1. Suppose on the contrary that $\sqrt{2} \in \mathbb{Q}$.
2. Then there exists n and d with $\sqrt{2} = n/d$, and this is a proper fraction (*i.e.* n and d have no common factors).

$\sqrt{2}$ is irrational

Theorem

The square root of 2 is not a rational number: $\sqrt{2} \notin \mathbb{Q}$.

Proof

1. Suppose on the contrary that $\sqrt{2} \in \mathbb{Q}$.
2. Then there exists n and d with $\sqrt{2} = n/d$, and this is a proper fraction (*i.e.* n and d have no common factors).
3. By definition $(n/d)^2 = n^2/d^2 = 2$.

$\sqrt{2}$ is irrational

Theorem

The square root of 2 is not a rational number: $\sqrt{2} \notin \mathbb{Q}$.

Proof

1. Suppose on the contrary that $\sqrt{2} \in \mathbb{Q}$.
2. Then there exists n and d with $\sqrt{2} = n/d$, and this is a proper fraction (*i.e.* n and d have no common factors).
3. By definition $(n/d)^2 = n^2/d^2 = 2$.
4. Thus n is divisible by 2. Write $n = 2n'$.

□

$\sqrt{2}$ is irrational

Theorem

The square root of 2 is not a rational number: $\sqrt{2} \notin \mathbb{Q}$.

Proof

1. Suppose on the contrary that $\sqrt{2} \in \mathbb{Q}$.
2. Then there exists n and d with $\sqrt{2} = n/d$, and this is a proper fraction (*i.e.* n and d have no common factors).
3. By definition $(n/d)^2 = n^2/d^2 = 2$.
4. Thus n is divisible by 2. Write $n = 2n'$.
5. We now have: $(2n')^2 = 4n'^2 = 2d^2$

□

$\sqrt{2}$ is irrational

Theorem

The square root of 2 is not a rational number: $\sqrt{2} \notin \mathbb{Q}$.

Proof

1. Suppose on the contrary that $\sqrt{2} \in \mathbb{Q}$.
2. Then there exists n and d with $\sqrt{2} = n/d$, and this is a proper fraction (*i.e.* n and d have no common factors).
3. By definition $(n/d)^2 = n^2/d^2 = 2$.
4. Thus n is divisible by 2. Write $n = 2n'$.
5. We now have: $(2n')^2 = 4n'^2 = 2d^2$
6. Thus d is also divisible by 2.

□

$\sqrt{2}$ is irrational

Theorem

The square root of 2 is not a rational number: $\sqrt{2} \notin \mathbb{Q}$.

Proof

1. Suppose on the contrary that $\sqrt{2} \in \mathbb{Q}$.
2. Then there exists n and d with $\sqrt{2} = n/d$, and this is a proper fraction (*i.e.* n and d have no common factors).
3. By definition $(n/d)^2 = n^2/d^2 = 2$.
4. Thus n is divisible by 2. Write $n = 2n'$.
5. We now have: $(2n')^2 = 4n'^2 = 2d^2$
6. Thus d is also divisible by 2.
7. This contradicts our assumption that there is a proper fraction to represent $\sqrt{2}$.

□

A set with no rational least upper bound

Theorem

The set $S = \{x \in \mathbb{Q} \mid x^2 \leq 2\}$ *has no rational least upper bound.*

Proof

The least upper bound is $\sqrt{2}$; but this is not a rational number.

$\square$

Theorem

The set $S = \{x \in \mathbb{R} \mid x^2 \leq 2\}$ *has a least upper bound:* $\sqrt{2}$.

The reals are uncountable

Theorem

The set of reals $\mathbb{R}$ is uncountable.

Proof of Cantor Diagonalization

1. Suppose on the contrary that $\mathbb{R}$ is countable.

□

The reals are uncountable

Theorem

The set of reals $\mathbb{R}$ is uncountable.

Proof of Cantor Diagonalization

1. Suppose on the contrary that $\mathbb{R}$ is countable.
2. Then there is an enumeration of the reals $\{x_0, x_1, \ldots\}$, and we express them as decimal expansions.

□

The reals are uncountable

Theorem

The set of reals $\mathbb{R}$ is uncountable.

Proof of Cantor Diagonalization

1. Suppose on the contrary that $\mathbb{R}$ is countable.
2. Then there is an enumeration of the reals $\{x_0, x_1, \ldots\}$, and we express them as decimal expansions.
3. Construct a new x which differs from x_n in the n-th digit.

□

The reals are uncountable

Theorem

The set of reals $\mathbb{R}$ is uncountable.

Proof of Cantor Diagonalization

1. Suppose on the contrary that $\mathbb{R}$ is countable.
2. Then there is an enumeration of the reals $\{x_0, x_1, \ldots\}$, and we express them as decimal expansions.
3. Construct a new x which differs from x_n in the n-th digit.
4. To avoid a problem with the equivalence of 9 recurring and 0, we change the digits d in the range $0 - 7$ to $d + 1$, and the digits 8 and 9 to 3 and 4 respectively.

□

The reals are uncountable

Theorem

The set of reals $\mathbb{R}$ is uncountable.

Proof of Cantor Diagonalization

1. Suppose on the contrary that $\mathbb{R}$ is countable.
2. Then there is an enumeration of the reals $\{x_0, x_1, \ldots\}$, and we express them as decimal expansions.
3. Construct a new x which differs from x_n in the n-th digit.
4. To avoid a problem with the equivalence of 9 recurring and 0, we change the digits d in the range $0 - 7$ to $d + 1$, and the digits 8 and 9 to 3 and 4 respectively.
5. Thus $x \neq x_n$, for any n, which is a contradiction.

□

The reals are uncountable

Theorem

The set of reals $\mathbb{R}$ is uncountable.

Proof of Cantor Diagonalization

1. Suppose on the contrary that $\mathbb{R}$ is countable.
2. Then there is an enumeration of the reals $\{x_0, x_1, \ldots\}$, and we express them as decimal expansions.
3. Construct a new x which differs from x_n in the n-th digit.
4. To avoid a problem with the equivalence of 9 recurring and 0, we change the digits d in the range $0 - 7$ to $d + 1$, and the digits 8 and 9 to 3 and 4 respectively.
5. Thus $x \neq x_n$, for any n, which is a contradiction.
6. Thus the reals are uncountable.

□

The reals are not computable

Theorem

There exist real numbers which cannot be represented on a computer.

Proof

There are countably many computer programs. There are uncountably many real numbers. Therefore some real numbers cannot be represented on any computer, no matter how much resource we have.

□

Computer Numbers

- There are two kinds of numbers provided by computer hardware: integer types and floating point types.

Computer Numbers

- There are two kinds of numbers provided by computer hardware: integer types and floating point types.
- Unfortunately neither of them is any of the systems we've already seen!

Integer Types and their errors

- The integer types are limited to a small number of bits — usually 32-bits or 64-bits, and the operations usually *wrap-around*.

Integer Types and their errors

- The integer types are limited to a small number of bits — usually 32-bits or 64-bits, and the operations usually *wrap-around*.
- A wrap-around occurs when an operation exceeds the number of bits available to hold the number; for example it usually comes as a surprise that in 32-bit signed integer:

$$2000000000 + 2000000000 = -294967296$$

Integer Types and their errors

- The integer types are limited to a small number of bits — usually 32-bits or 64-bits, and the operations usually *wrap-around*.
- A wrap-around occurs when an operation exceeds the number of bits available to hold the number; for example it usually comes as a surprise that in 32-bit signed integer:

$$2000000000 + 2000000000 = -294967296$$

- This behaviour is the cause of many bugs in computer programs!

Floating Point Types and their errors

- Each floating point type is a subset of the rational numbers $\mathbb{Q}$.

Floating Point Types and their errors

- Each floating point type is a subset of the rational numbers $\mathbb{Q}$.
- The most frequently encountered types are `float` with 23-bit mantissa, and 8-bit exponent, and `double` with 53-bit mantissa, and 10-bit exponent.

Floating Point Types and their errors

- Each floating point type is a subset of the rational numbers $\mathbb{Q}$.
- The most frequently encountered types are `float` with 23-bit mantissa, and 8-bit exponent, and `double` with 53-bit mantissa, and 10-bit exponent.
- It often comes as a surprise that floating point numbers do not have an associative addition operation, *i.e.* it is sometimes possible that:

$$(x + y) + z \neq x + (y + z)$$

Floating Point Types and their errors

- Each floating point type is a subset of the rational numbers $\mathbb{Q}$.
- The most frequently encountered types are `float` with 23-bit mantissa, and 8-bit exponent, and `double` with 53-bit mantissa, and 10-bit exponent.
- It often comes as a surprise that floating point numbers do not have an associative addition operation, *i.e.* it is sometimes possible that:

$$(x + y) + z \neq x + (y + z)$$

- To see this, take $x = 1e30$, $y = -x$, and $z = 1e - 30$.

Floating Point Types and their errors

- Each floating point type is a subset of the rational numbers $\mathbb{Q}$.
- The most frequently encountered types are `float` with 23-bit mantissa, and 8-bit exponent, and `double` with 53-bit mantissa, and 10-bit exponent.
- It often comes as a surprise that floating point numbers do not have an associative addition operation, *i.e.* it is sometimes possible that:

$$(x + y) + z \neq x + (y + z)$$

- To see this, take $x = 1e30$, $y = -x$, and $z = 1e - 30$.
- Then $x + y = 0$ and $y + z = y$; the second result occurring because there insufficient bits in the mantissa to hold all bits accurately.

Floating Point Types and their errors

- Each floating point type is a subset of the rational numbers $\mathbb{Q}$.
- The most frequently encountered types are `float` with 23-bit mantissa, and 8-bit exponent, and `double` with 53-bit mantissa, and 10-bit exponent.
- It often comes as a surprise that floating point numbers do not have an associative addition operation, *i.e.* it is sometimes possible that:

$$(x + y) + z \neq x + (y + z)$$

- To see this, take $x = 1e30$, $y = -x$, and $z = 1e - 30$.
- Then $x + y = 0$ and $y + z = y$; the second result occurring because there insufficient bits in the mantissa to hold all bits accurately.
- This behaviour is *cancellation*, and is also the cause of many bugs in computer programs!

A practical problem with Floating Point

- Consider calculating a dot-product between two vectors of `double` floating point numbers.

A practical problem with Floating Point

- Consider calculating a dot-product between two vectors of `double` floating point numbers.
- Naïvely, we could multiply the numbers pairwise and then sum them.

A practical problem with Floating Point

- Consider calculating a dot-product between two vectors of `double` floating point numbers.
- Naïvely, we could multiply the numbers pairwise and then sum them.
- The problem with doing this is that we could get the same cancellation effect we saw previously.

A practical problem with Floating Point

- Consider calculating a dot-product between two vectors of `double` floating point numbers.
- Naïvely, we could multiply the numbers pairwise and then sum them.
- The problem with doing this is that we could get the same cancellation effect we saw previously.
- To solve this problem we could sort the pairwise terms into ascending order by their absolute value.

A practical problem with Floating Point

- Consider calculating a dot-product between two vectors of `double` floating point numbers.
- Naïvely, we could multiply the numbers pairwise and then sum them.
- The problem with doing this is that we could get the same cancellation effect we saw previously.
- To solve this problem we could sort the pairwise terms into ascending order by their absolute value.
- This makes dot-product (and therefore matrix multiplication) much more expensive.

Numerical Analysis

- The discipline of *Numerical Analysis* was invented by Wilkinson when it became apparent that early numeric computer programs on the Manchester Mark 1 were delivering wrong answers!

Numerical Analysis

- The discipline of *Numerical Analysis* was invented by Wilkinson when it became apparent that early numeric computer programs on the Manchester Mark 1 were delivering wrong answers!
- I usually advise my computer science students that they are not qualified to use floating point types.

Numerical Analysis

- The discipline of *Numerical Analysis* was invented by Wilkinson when it became apparent that early numeric computer programs on the Manchester Mark 1 were delivering wrong answers!
- I usually advise my computer science students that they are not qualified to use floating point types.
- Usually numerical analysts are not experts in software engineering either, which is something of a conundrum!

Are the computable reals an answer?

- Can we use the computable reals instead of the reals?

Are the computable reals an answer?

- Can we use the computable reals instead of the reals?
- Using computable real implementations such as ERA, we *can* add, subtract, multiply divide by non-zero quantities, take exponentials and logs (of positive quatities), *etc*.

Are the computable reals an answer?

- Can we use the computable reals instead of the reals?
- Using computable real implementations such as ERA, we *can* add, subtract, multiply divide by non-zero quantities, take exponentials and logs (of positive quatities), *etc*.
- The problem with the computable reals is that there is no comparison operation: we cannot determine in a finite time whether $x > 0$, or more generally whether $x > y$.

Hybrid Systems

- A hybrid dynamical system is one in which there are a mixture of interacting continuously varying real-valued variables interacting with discrete boolean signals.

Hybrid Systems

- A hybrid dynamical system is one in which there are a mixture of interacting continuously varying real-valued variables interacting with discrete boolean signals.
- Clearly a system of very simple neurons communicating with spikes is a hybrid system.

Hybrid Systems

- A hybrid dynamical system is one in which there are a mixture of interacting continuously varying real-valued variables interacting with discrete boolean signals.
- Clearly a system of very simple neurons communicating with spikes is a hybrid system.
- In general such a system *cannot* be computable, as there is a need to determine when a membrane voltage has exceeded it's threshold.

Hybrid Systems

- A hybrid dynamical system is one in which there are a mixture of interacting continuously varying real-valued variables interacting with discrete boolean signals.
- Clearly a system of very simple neurons communicating with spikes is a hybrid system.
- In general such a system *cannot* be computable, as there is a need to determine when a membrane voltage has exceeded it's threshold.
- Unless a hybrid system is degenerate (*i.e.* is entirely continuous or entirely discrete) it will almost inevitably be chaotic.

Hybrid Systems

- A hybrid dynamical system is one in which there are a mixture of interacting continuously varying real-valued variables interacting with discrete boolean signals.
- Clearly a system of very simple neurons communicating with spikes is a hybrid system.
- In general such a system *cannot* be computable, as there is a need to determine when a membrane voltage has exceeded it's threshold.
- Unless a hybrid system is degenerate (*i.e.* is entirely continuous or entirely discrete) it will almost inevitably be chaotic.
- A chaotic system is one in which the final state depends on an excessively precise initial state.

Hybrid Systems

- A hybrid dynamical system is one in which there are a mixture of interacting continuously varying real-valued variables interacting with discrete boolean signals.
- Clearly a system of very simple neurons communicating with spikes is a hybrid system.
- In general such a system *cannot* be computable, as there is a need to determine when a membrane voltage has exceeded it's threshold.
- Unless a hybrid system is degenerate (*i.e.* is entirely continuous or entirely discrete) it will almost inevitably be chaotic.
- A chaotic system is one in which the final state depends on an excessively precise initial state.
- One quick way to gauge whether a system might be chaotic is to make small pertubations to the initial condition, and see if there is a large effect on the output.

Philosophical Issues

- **So is it still possible to model a brain using a computer simulation?**

Philosophical Issues

- **So is it still possible to model a brain using a computer simulation?**
- Yes. The limits on computability are simply a reminder that the simulation is an *approximate* model of the system.

Philosophical Issues

- **So is it still possible to model a brain using a computer simulation?**
- Yes. The limits on computability are simply a reminder that the simulation is an *approximate* model of the system.
- **Can physical models like BrainScales solve this problem?**

Philosophical Issues

- **So is it still possible to model a brain using a computer simulation?**
- Yes. The limits on computability are simply a reminder that the simulation is an *approximate* model of the system.
- **Can physical models like BrainScales solve this problem?**
- There is a similar problem making accurate comparator circuits, so these systems too, are also approximate models of the system.

Conclusion

At the end of this lecture you will:

- Be aware of different sorts of mathematical arithemtic systems.

Conclusion

At the end of this lecture you will:

- Be aware of different sorts of mathematical arithemtic systems.
- Have a feeling for countability arguments.

Conclusion

At the end of this lecture you will:

- Be aware of different sorts of mathematical arithemtic systems.
- Have a feeling for countability arguments.
- Be aware of the practical problems that occur when using the usual numbers provided on a computer.

Conclusion

At the end of this lecture you will:

- Be aware of different sorts of mathematical arithemtic systems.
- Have a feeling for countability arguments.
- Be aware of the practical problems that occur when using the usual numbers provided on a computer.
- Be aware of the issues and limitations arising when using numeric simulations.

Conclusion

At the end of this lecture you will:

- Be aware of different sorts of mathematical arithemtic systems.
- Have a feeling for countability arguments.
- Be aware of the practical problems that occur when using the usual numbers provided on a computer.
- Be aware of the issues and limitations arising when using numeric simulations.
- Be alert to the issue of chaotic behaviour, which we might expect to be common in brain simulation.