Errors

Intensive Computation

Annalisa Massini
2018/2019

References

- Scientific Computing: An Introductory Survey - Chapter 1 – M.T. Heath
  http://heath.cs.illinois.edu/scicomp/notes/index.html

- Principles of Scientific Computing - Sources of Error
  http://www.cs.nyu.edu/courses/spring09/G22.2112-001/book/SourcesOfError.chapter.pdf

- Error analysis for linear systems
  http://homepage.math.uiowa.edu/~atkinson/m171.dir/sec_8-4.pdf

OVERVIEW

Sources of Approximation

- ***Before*** computation
  - modeling
  - empirical measurements
  - previous computations
- ***During*** computation
  - truncation or discretization
  - rounding
- **Accuracy** of final result reflects all these
- **Uncertainty** in input may be amplified by the problem
- **Perturbations** during computation may be *amplified* by algorithm

Approximations: Example

- Computing surface area of Earth using formula $A = 4\pi r^2$ involves ***several approximations***

Approximations: Example

- Computing surface area of Earth using formula $A = 4\pi r^2$ involves ***several approximations***
  - Earth is modeled as **sphere**, idealizing its true shape

Approximations: Example

- Computing surface area of Earth using formula $A = 4\pi r^2$ involves ***several approximations***
  - Earth is modeled as **sphere**, idealizing its true shape
  - Value for **radius** is based on empirical measurements and previous computations

Approximations: Example

- Computing surface area of Earth using formula $A = 4\pi r^2$ involves ***several approximations***
  - Earth is modeled as **sphere**, idealizing its true shape
  - Value for **radius** is based on empirical measurements and previous computations
  - Value for π requires **truncating** infinite process

Approximations: Example

- Computing surface area of Earth using formula $A = 4\pi r^2$ involves ***several approximations***
  - Earth is modeled as **sphere**, idealizing its true shape
  - Value for **radius** is based on empirical measurements and previous computations
  - Value for π requires **truncating** infinite process
  - Values for input data and results of arithmetic operations are **rounded** in computer

Sources of Error

- Scientific computing usually gives ***inexact answers***

Example

- the code x = sqrt(2) produces something that is not the mathematical $\sqrt{2}$
- x differs from $\sqrt{2}$ by an amount that we call the **error**

Sources of Error

- The **goal** of a **scientific computation** is rarely the exact answer, but a **result** that is **as accurate as needed**

- An **accurate result** has a **small error**

- We use
  - A to denote the exact answer to some problem
  - $\widehat{A}$ to denote the computed approximation

- The **error** is $\widehat{A} - A$

Sources of Error

- There are four primary ways in which error is introduced into a computation:

- ***Roundoff error*** from inexact computer arithmetic
- ***Truncation error*** from approximate formulas
- ***Termination of iterations***
- ***Statistical error*** in Monte Carlo

Sources of Error

- It is important:
  - To understand the likely ***relative sizes*** **of the** ***various kinds of error***
  - To focus our efforts on ***reducing the largest sources of error***
- This will help:
  - To better design computational algorithms
  - To understand the various ***sources of error*** and to debug scientific computing software
- Finally, if a result is supposed to be A and instead is $\hat{A}$, it is important to understand if:
  - The difference between A and $\hat{A}$ is the result of a *programming mistake*
  - **Or** the way of calculating something is simply *not accurate enough*

Error propagation

- A typical computation has several stages, with the results of one stage being the inputs to the next
- *Errors* in the output of *one stage* most likely mean that the *output of the next* would be *inexact* even if the second stage computations were done exactly
- **It is unlikely that the second stage would produce the *exact output from inexact inputs***
- On the contrary, it is possible to have ***error amplification*** due to ***error propagation***

Error propagation

- If the second stage output is very ***sensitive*** to its input, small errors in the input could result in large errors in the output that is, the error will be amplified
- A method with large error amplification is ***unstable***
- The **condition number** of a problem measures the ***sensitivity*** of the answer to small changes in its input data
- The ***condition number*** is ***determined by the problem***, not the method used to solve it
- The *accuracy of a solution is limited by the condition number* of the problem

Error propagation

- A problem is called ***ill-conditioned*** if the condition number is so large that it is hard/impossible to solve it accurately enough

- A computational strategy is likely to be **unstable** if it has an *ill-conditioned* ***subproblem***

Example

- Suppose we solve a system of linear differential equations using the eigenvector basis of the corresponding matrix
- Finding eigenvectors of a matrix can be *ill-conditioned*
- This makes the eigenvector approach to solving linear differential equations potentially *unstable*, even when the differential equations themselves are well-conditioned

COMPUTATIONAL ERRORS

Absolute Error and Relative Error

- **absolute error**: approximate value - true value
- **relative error**: $\frac{\text{absolute error}}{\text{true value}}$

- Equivalently: approx value = (true value) x (1 + rel error)

- ***True value*** is usually ***unknown***, so we ***estimate*** or ***bound*** the error rather than to compute it exactly

- The *relative error* is often taken *relative to approximate value*, rather than (unknown) true value

Data Error and Computational Error

Typical problem: compute value of function $f: R \rightarrow R$ for given argument

- x = true value of input
- $f(x)$ = desired result
- $\hat{x}$ = approximate (inexact) input
- $\hat{f}$ = approximate function actually computed

- Total error: $\hat{f}(\hat{x}) - f(x) =$

$$\underbrace{\hat{f}(\hat{x}) - f(\hat{x})}_{\text{computational error}} + \underbrace{f(\hat{x}) - f(x)}_{\text{propagated data error}}$$

computational error + **propagated data error**

- ***Algorithm has no effect on propagated data error***

Example

- We need to compute **value of $\sin(\pi/8)$** without a calculator
- We could approximate:
  - π with 22/7, but it is easier to use $\pi = 3$
- Also, we can approximate
  - **sine** function by truncating Taylor series after first term (we are considering a small value of the argument), that is $\sin x = x$

Example

- **Computational error** is obtained by

  $\hat{f}(\hat{x}) \approx \sin(3/8) \approx 3/8 \approx 0.3750$

  $f(\hat{x}) = 0.3662$ (obtained by using a calculator)

- Hence the **computational error** is

  $\hat{f}(\hat{x}) - f(\hat{x}) = 0.3750 - 0.3662 = \mathbf{0.0088}$

- **Propagated data error** is obtained by

  $f(\hat{x}) = 0.3826$ (obtained by using a calculator)

- Hence the **Propagated data error** is

  $f(\hat{x}) - f(x) = 0.3662 - 0.3826 = \mathbf{-0.0164}$

- Computational error and propagated data error balance each other

Truncation Error and Rounding Error

- **Truncation error**: difference between true result (for actual input) and result produced by given algorithm using exact arithmetic
  - Due to approximations such as truncating infinite series or terminating iterative sequence before convergence
- **Rounding error**: difference between result produced by given algorithm using exact arithmetic and result produced by same algorithm using limited precision arithmetic
  - Due to inexact representation of real numbers and arithmetic operations upon them
- **Computational error** is the sum of **truncation error** and **rounding error**, but ***one of these usually dominates***

Example: Finite Difference Approximation

- Error in finite difference approximation

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

exhibits tradeoff between rounding error and truncation error

- ***Truncation error*** bounded by $\frac{Mh}{2}$, where M bounds $|f''(t)|$ for t near x

Example: Finite Difference Approximation

- Error in finite difference approximation

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

exhibits tradeoff between rounding error and truncation error

- ***Truncation error*** bounded by $\frac{Mh}{2}$, where M bounds $|f''(t)|$ for t near x

- ***Rounding error*** bounded by $\frac{2\varepsilon}{h}$, where error in function values bounded by ε

Example: Finite Difference Approximation

- Error in finite difference approximation

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

exhibits tradeoff between rounding error and truncation error

- ***Truncation error*** bounded by $\frac{Mh}{2}$
- ***Rounding error*** bounded by $\frac{2\varepsilon}{h}$
- **Computational error** is: $\frac{Mh}{2} + \frac{2\varepsilon}{h}$
  - It increases for ***smaller*** h because of rounding error
  - It increases for ***larger*** h because of truncation error

Example: Finite Difference Approximation

- Error in finite difference approximation

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

exhibits tradeoff between rounding error and truncation error

- ***Truncation error*** bounded by $\frac{Mh}{2}$
- ***Rounding error*** bounded by $\frac{2\varepsilon}{h}$
- **Computational error** is: $\frac{Mh}{2} + \frac{2\varepsilon}{h}$
- Total error minimized when $h \approx 2\sqrt{\varepsilon/M}$

FORWARD ERRORS AND BACKWARD ERRORS

Forward and Backward Error

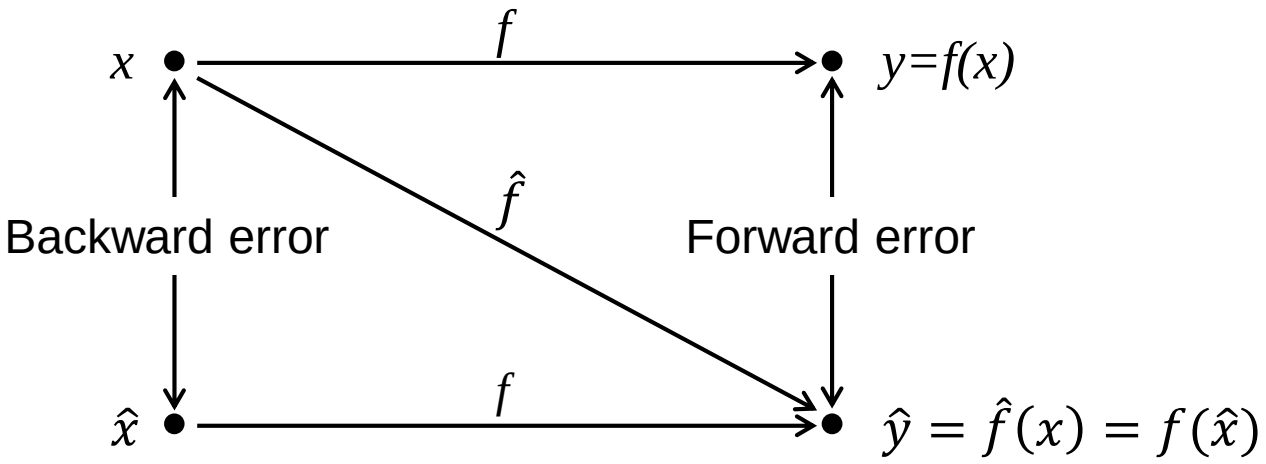
- Suppose we want to compute $y = f(x)$, where $f : R \rightarrow R$
- Instead we obtain the **approximate value $\hat{y}$**
  - **Forward error**: $\Delta y = \hat{y} - y$

Forward and Backward Error

- Suppose we want to compute $y = f(x)$, where $f : R \rightarrow R$
- Instead we obtain the **approximate value $\hat{y}$**
  - **Forward error**: $\Delta y = \hat{y} - y$
  - **Backward error**: $\Delta x = \hat{x} - x$, where $f(\hat{x}) = \hat{y}$

Forward and Backward Error

- Suppose we want to compute $y = f(x)$, where $f : R \rightarrow R$
- Instead we obtain the **approximate value $\hat{y}$**
  - **Forward error**: $\Delta y = \hat{y} - y$
  - **Backward error**: $\Delta x = \hat{x} - x$, where $f(\hat{x}) = \hat{y}$



Example: Forward and Backward Error

As approximation to $y = \sqrt{2}$, the value $\hat{y} = 1.4$ has:

- *absolute* forward error

$$|\Delta y| = |\hat{y} - y| = |1.4 - 1.41421 \ldots| \approx 0.0142$$

or *relative forward error* of about 1%

Example: Forward and Backward Error

As approximation to $y = \sqrt{2}$, the value $\hat{y} = 1.4$ has:

- *absolute* **forward** error

$$|\Delta y| = |\hat{y} - y| = |1.4 - 1.41421\ldots| \approx 0.0142$$

  or *relative forward error* of about 1%

- Since $\sqrt{1.96} = 1.4$, *absolute* **backward** error is

$$|\Delta x| = |\hat{x} - x| = |1.96 - 2| = 0.04$$

  or *relative backward error* of 2%

Backward Error Analysis

- **Idea of the backward error analysis** →

  approximate solution is *exact solution to modified problem*

- Questions are:
  - How much must *original problem* change to give result actually obtained?
  - How much data error in input would explain *all* the errors in computed result?

- *Approximate solution* is **good** if it is *exact solution* to *nearby* problem

- Backward error is often ***easier*** to estimate than forward error

Example: Backward Error Analysis

- Approximating cosine function $f(x) = \cos x$, by truncating Taylor series after two terms gives

$$\hat{y} = \hat{f}(x) = 1 - x^2/2$$

- ***Forward error*** is given by

$$\Delta y = \hat{y} - y = \hat{f}(x) - f(x) = 1 - x^2/2 - \cos x$$

- To determine ***backward error***, we need value $\hat{x}$ such that

$$f(\hat{x}) = \hat{f}(x)$$

- For cosine function, $\hat{x} = \arccos\left(\hat{f}(x)\right) = \arccos(\hat{y})$

Example continued

- For $x = 1$

$$y = f(1) = \cos(1) \approx 0.5403$$

$$\hat{y} = \hat{f}(1) = 1 - \frac{1^2}{2} = 0.5$$

$$\hat{x} = \arccos(\hat{y}) = \arccos(0.5) = 1.0472$$

- **Forward error**: $\Delta y = \hat{y} - y \approx 0.5 - 0.5403 = -0.0403$
- **Backward error**: $\Delta x = \hat{x} - x \approx 1.0472 - 1 = 0.0472$

SENSITIVITY AND CONDITIONING

Well-Posed Problems

- Problem is well-posed if solution
  - exists
  - is unique
  - depends continuously on problem data

  Otherwise, problem is ill-posed

- Even if problem is well posed, solution may still be sensitive to input data

- Computational algorithm should not make sensitivity worse

Sensitivity and Conditioning

- Problem is **insensitive**, or **well-conditioned**, if relative change in input causes similar relative change in solution

- Problem is **sensitive**, or **ill-conditioned**, if relative change in solution can be much larger than that in input data

- ***Condition number*** :

$$\text{cond} = \frac{|\text{relative change in solution}|}{|\text{relative change in input data}|}$$

$$= \frac{|(f(\hat{x}) - f(x))/f(x)|}{|(\hat{x} - x)/x|} = \frac{|\Delta y/y|}{|\Delta x/x|}$$

- Problem is sensitive, or ill-conditioned, if $\mathbf{cond} \gg \mathbf{1}$

Condition number

- Condition number is **amplification factor** relating *relative forward error* to *relative backward error*

$$|\text{relative forward error}| = \text{cond} \times |\text{relative backward error}|$$

- Condition number usually is not known exactly and may vary with input, so rough estimate or upper bound is used for cond, yielding

$$|\text{relative forward error}| < \text{cond} \times |\text{relative backward error}|$$

Example: Evaluating Function

- Evaluating function f for approximate input $\hat{x} = x + \Delta x$ instead of true input x gives
- **Absolute forward error**: $\hat{x} = f(x + \Delta x) - f(x) \approx f'(x)\Delta x$
- **Relative forward error**: $\frac{f(x+\Delta x)-f(x)}{f(x)} \approx \frac{f'(x)\Delta x}{f(x)}$
- **Condition number**: $\text{cond} \approx \left| \frac{\frac{f'(x)\Delta x}{f(x)}}{\frac{\Delta x}{x}} \right| = \left| \frac{x f'(x)}{f(x)} \right|$
- Relative error in function value can be much larger or smaller than that in input, depending on particular f and x

Example: Sensitivity

- Tangent function is ***sensitive*** for arguments near $\pi/2$
  - $\tan(1.57079) \approx 1.58058 \times 10^5$
  - $\tan(1.57078) \approx 6.12490 \times 10^4$
- Relative change in output is quarter million times greater than relative change in input
  - For $x = 1.57079$, $cond \approx 2.48275 \times 10^5$

Stability

- Algorithm is **stable** if result produced is relatively insensitive to perturbations during computation
- Stability of algorithms is analogous to conditioning of problems
- From point of view of backward error analysis, algorithm is stable if result produced is exact solution to nearby problem
- For stable algorithm, effect of computational error is no worse than effect of small data error in input

Accuracy

- **Accuracy**: closeness of computed solution to true solution of problem
- Stability alone does not guarantee accurate results
- Accuracy depends on *conditioning of problem* as well as *stability of algorithm*
- *Inaccuracy* can result from applying:
  - stable algorithm to ill-conditioned problem or
  - unstable algorithm to well-conditioned problem
- Applying stable algorithm to well-conditioned problems yields **accurate solution**

Condition Number for Linear Systems

- Consider the **system**

$$7x + 10y = 1$$

$$5x + 7y = 0.7$$

- It has **solution** $x = 0 \qquad y = 0.1$

- The **perturbed system**

$$7\hat{x} + 10\hat{y} = 1.01$$

$$5\hat{x} + 7\hat{y} = 0.69$$

has the **solution** $\hat{x} = -0.17 \qquad \hat{y} = 0.22$

Condition Number for Linear Systems

- In solving a linear system $Ax = b$, we need to know the **sensitivity** of the solution x to changes in the right side b
- Consider the two linear systems $Ax = b$ $\quad A\tilde{x} = b + r$
- What is $\dfrac{\|\tilde{x} - x\|}{\|x\|}$?
- We simply solve for $\tilde{x} - x$

$$\tilde{x} - x = A^{-1}[b + r] - A^{-1}b = A^{-1}r \quad \rightarrow \quad \|\tilde{x} - x\| = \|A^{-1}r\| \leq \|A^{-1}\| \, \|r\|$$

$$\frac{\|\tilde{x} - x\|}{\|x\|} \leq \frac{\|A^{-1}\| \, \|r\|}{\|x\|} = \|A^{-1}\| \, \|A\| \frac{\|r\|}{\|A\| \, \|x\|}$$

Condition Number for Linear Systems

- Since $Ax = b$, we have $\|b\| \leq \|A\| \, \|x\|$, and then

$$\frac{\|\tilde{x} - x\|}{\|x\|} \leq \|A^{-1}\| \, \|A\| \frac{\|r\|}{\|b\|}$$

- The number $\text{cond}(A) = \|A^{-1}\| \, \|A\|$ is the **condition number** for the matrix A and for the linear system

$$\frac{\|\tilde{x} - x\|}{\|x\|} \leq \text{cond}(A) \frac{\|r\|}{\|b\|}$$

Condition Number for Linear Systems

- We can also prove a ***lower inequality***, obtaining

$$\frac{1}{\text{cond}(A)}\frac{\|r\|}{\|b\|} \leq \frac{\|\tilde{x}-x\|}{\|x\|} \leq \text{cond}(A)\frac{\|r\|}{\|b\|}$$

- In addition, given any nonsingular A, there are vectors b and r for which either of the above *inequalities are actually an equalities*

Condition Number for Linear Systems

- With $\frac{1}{\text{cond}(A)} \frac{\|r\|}{\|b\|} \leq \frac{\|\tilde{x} - x\|}{\|x\|} \leq \text{cond}(A) \frac{\|r\|}{\|b\|}$

  we see that

  - if **$\text{cond}(A) \approx 1$**, then small 'relative' changes in b are guaranteed to lead to equally small 'relative' changes in x
  - if **$\text{cond}(A)$ is very large**, then there are values of b and r for which $\frac{\|r\|}{\|b\|}$ is small and $\frac{\|\tilde{x} - x\|}{\|x\|}$ is large

- In practice, it is ***very difficult*** to know whether your choice of b and r is good or bad

FLOATING POINT NUMBERS

Floating-Point Numbers

- **Floating-point numbers** are designated as follows
  - Sign
  - Exponent
  - Mantissa
  - Sign, exponent, and mantissa are stored in separate fixed-width fields of each floating-point word
- IEEE floating-point systems are:
  - Almost universal in digital computers and supported by all major CPUs
  - IEEE Standard 754 established in 1985 as uniform standard for floating point arithmetic
    - Before that, many idiosyncratic formats
  - Two representations
    - Single precision (32-bit)
    - Double precision (64-bit)

Floating-Point Numbers

- Encoding

  - Single precision: 32 bits total
    - 8 `exp` bits,
    - 23 `mantissa` bits
  - Double precision: 64 bits total
    - 11 `exp` bits,
    - 52 `mantissa` bits

  - Extended precision: 80 bits
    - 15 `exp` bits, 63 `frac` bits
    - 1 bit wasted - only found in Intel-compatible machines

Floating-Point Numbers

- Encoding

  - **Mantissa** (*Significand*)
    - Is assumed to be 1.xxxxx
      - So a mantissa equal to 000…0 is interpreted to be 1.0, and a mantissa equal to 11…11 is interpreted to be 1.1111
    - Always has a leading pre-binary-point 1 bit, so ***no need to represent it explicitly*** (hidden bit then restored)
    - This is called a normalized representation
    - Special cases are used to represent denormalized mantissas, NaN, etc
  - **Exponent**: excess representation → actual exponent + Bias
    - Ensures exponent is unsigned
    - *Single*: Bias = 127 *Double*: Bias = 1023

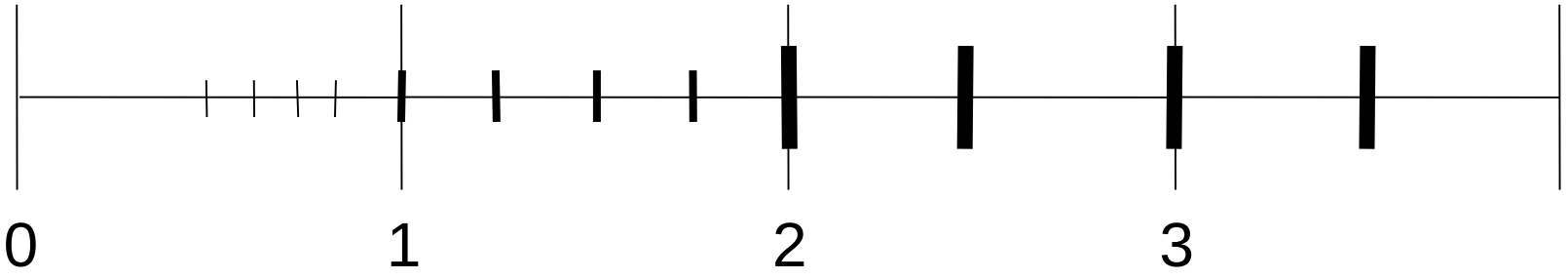
Floating-Point Numbers

- Not all real numbers ***exactly*** representable
- Represented real numbers are called machine numbers

- Floating-point systems look grainy and unequally spaced

- Example
  - 3 bit mantissa
  - exponent {-1,0,1}

e = -1	e = 0	e = 1
1.00 X 2^(-1) = 1/2	1.00 X 2^0 = 1	1.00 X 2^1 = 2
1.01 X 2^(-1) = 5/8	1.01 X 2^0 = 5/4	1.01 X 2^1 = 5/2
1.10 X 2^(-1) = 3/4	1.10 X 2^0 = 3/2	1.10 X 2^1= 3
1.11 X 2^(-1) = 7/8	1.11 X 2^0 = 7/4	1.11 X 2^1 = 7/2



Floating-Point Numbers

- If a real number x is not exactly representable, then it is approximated by “nearby” floating-point number
- Two commonly used rules:
  - **chop**: truncate x after (p - 1)st digit - also called *round toward zero*
  - **round to nearest**: fl(x) is nearest floating-point number to x, using floating-point number whose last stored digit is even in case of tie; also called *round to even*
- This process is called rounding
  - Error introduced is called **rounding error**
  - *Round to nearest is most accurate*, and is default rounding rule in IEEE systems

Floating-Point Numbers

- Accuracy of floating-point system characterized by unit roundoff or machine precision (or machine epsilon) ε_{mach}
- Maximum relative error in representing real number x within range of floating-point system is given by

$$\left|\frac{fl(x) - x}{x}\right| \leq \varepsilon_{mach}$$

- For IEEE floating-point systems
  - $\varepsilon_{mach} = 2^{-24} \approx 10^{-7}$ in single precision
  - $\varepsilon_{mach} = 2^{-53} \approx 10^{-16}$ in double precision
  - So IEEE single and double precision systems have about 7 and 16 decimal digits of precision, respectively

FP Multiplication

- Operands

  $(-1)^{s1}\ M1\ \ 2^{E1}$ $(-1)^{s2}\ M2\ \ 2^{E2}$

- Exact Result

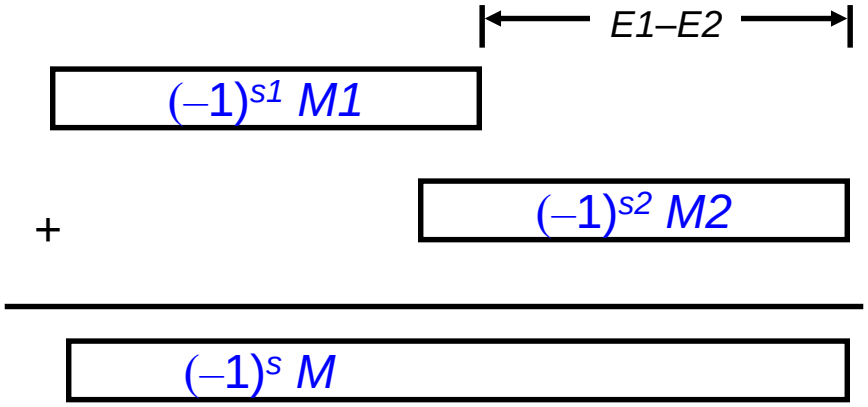
  $(-1)^{s}\ M\ \ 2^{E}$

  - Sign *s*: *s1 ^ s2*
  - Significand *M*: *M1 * M2*
  - Exponent *E*: *E1 + E2*

- Fixing
  - If $M \geq 2$, shift *M* right, increment *E*
  - ***Overflow*** if *E* out of range,
  - Round *M* to fit precision

FP Addition

- Operands
  - $(-1)^{s1}\ M1\ \ 2^{E1}$
  - $(-1)^{s2}\ M2\ \ 2^{E2}$
    - Assume $E1 > E2$
- Exact Result
  - $(-1)^{s}\ M\ \ 2^{E}$
    - Sign s, significand M:
      - Result of signed align & add
    - Exponent E: $E1$



- Fixing
  - If $M \geq 2$, shift M right, increment E
  - if $M < 1$, shift M left k positions, decrement E by k
  - ***Overflow*** if E out of range
  - Round M to fit precision

Floating-Point Arithmetic

- Addition or subtraction: Shifting of mantissa to make exponents match may cause loss of some digits of smaller number, possibly all of them
- Multiplication: Product of two p-digit mantissas contains up to $2p$ digits, so result may not be representable
- Division: Quotient of two p-digit mantissas may contain more than p digits
- Result of floating-point arithmetic operation may differ from result of corresponding real arithmetic operation on same operands

Floating-Point Arithmetic

Example: cancellation

- **Subtraction** between two *p*-digit numbers having same sign and similar magnitudes yields result with fewer than p digits
- Reason is that leading digits of two numbers cancel
- Example: $1.92403 \times 10^{2} - 1.92275 \times 10^{2} = 1.28000 \times 10^{-1}$
- Result is correct, and exactly representable but has only three significant digits
- Despite exactness of result, cancellation often implies serious loss of information

Floating-Point Arithmetic

Cancellation

- Operands are often uncertain due to rounding or other previous errors, so relative uncertainty in difference may be large

- Example
  - if ε is positive floating-point number slightly smaller than ε_{mach}, then (1 + ε) - (1 - ε) = 1 - 1 = 0 in floating-point arithmetic
  - it is correct for actual operands of final subtraction, but true result of overall computation, 2 ε, has been completely lost

- Subtraction itself is not at fault: it merely signals loss of information that had already occurred

Floating-Point Arithmetic

Cancellation

- Digits lost to ***cancellation*** are *most significant*, leading digits
- Digits lost in ***rounding*** are *least significant*

- Because of this effect, it is generally bad idea to compute any small quantity as difference of large quantities, since rounding error is likely to dominate result

- For example, summing alternating series, such as

$$e^x = 1 + x + x^2/2! + x^3/3! + \ldots$$

for $x < 0$, may give disastrous results due to catastrophic cancellation

MEASUREMENT ERRORS

Averaging, Errors and Uncertainty - Upenn - Lab Manual - Physics & Astronomy Dept

https://www.physics.upenn.edu/sites/www.physics.upenn.edu/files/Managing%20Errors%20and%20Uncertainty.pdf

Measurements Errors

There are three types of limitations to measurements:

1) **Instrumental limitations**

- Any measuring device can only be used to measure to with a certain degree of fineness
- Our measurements are no better than the instruments we use to make them

Measurements Errors

There are three types of limitations to measurements:

2) **Systematic errors and blunders**

- These are caused by a mistake *which does not change* during the measurement.
- For example, if the platform balance you used to weigh something was not correctly set to zero with no weight on the pan, all your subsequent measurements of mass would be too large.

- Systematic errors do not enter into the uncertainty. They are either identified and eliminated or lurk in the background producing a shift from the true value.

Measurements Errors

There are three types of limitations to measurements:

3) **Random errors**

- These arise from unnoticed variations in measurement technique, tiny changes in the experimental environment, etc.
- Random variations affect precision. Truly random effects average out if the results of a large number of trials are combined.

Precision and Accuracy

- A **precise** measurement is one where independent measurements of the same quantity *closely cluster* about a single value that may or may not be the correct value

- An **accurate** measurement is one where independent measurements *cluster about the true value* of the measured quantity

Systematic errors:

- are *not random* and therefore can never cancel out
- affect the ***accuracy*** but not the ***precision*** of a measurement

Precision and Accuracy

- A. Low-precision, Low-accuracy:

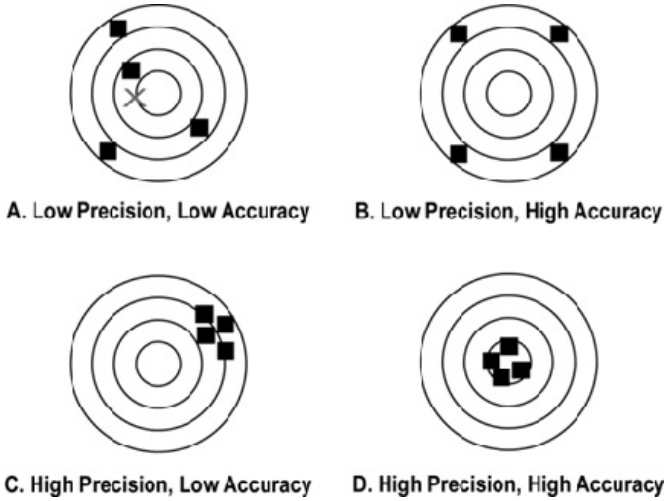
  The average (the X) is not close to the center

- B. Low-precision, High-accuracy:

  The average is close to the true value

- C. High-precision, Low-accuracy:

  The average is not close to the true value



Uncertainty of Measurements

- Errors are quantified by associating an **uncertainty** with each measurement

- **Example**
  - The best estimate of a length L is 2.59 cm,
  - Due to uncertainty, the length might be:
    - as small as 2.57cm
    - or as large as 2.61cm

Uncertainty of Measurements

- Length L can be expressed with its uncertainty in two different ways:

1. **Absolute Uncertainty**

   Expressed in the units of the measured quantity:

   $$L = 2.59 \pm 0.02 \text{ cm}$$

2. **Percentage Uncertainty**

   Expressed as a percentage independent of the units above, since $0.02 / 2.59 \approx 1\%$

   we would write $L = 2.59 \text{ cm} \pm 1\%$

Significant Digits

Experimental numbers must be written in a way consistent with the precision to which they are known: **significant digits** (or **figures**) that have physical meaning

1. All definite digits and the first doubtful digit are considered significant
2. Leading zeros are **not** significant digits

   Example: L=2.31 cm has 3 significant figures.

   For L=0.0231 m, the zeros serve to move the decimal point to the correct position
3. Trailing zeros are significant digits: they indicate the number's precision
4. One significant digit should be used to report the uncertainty or occasionally two, especially if the second digit is a five

Rounding Numbers

- To keep the correct number of significant figures, numbers must be rounded off
- The discarded digit is called the **remainder**
- There are three rules for rounding:
  - **Rule 1**: If the remainder is less than 5, drop the last digits

    Rounding to one decimal place: 5.346 → 5.3
  - **Rule 2**: If the remainder is greater than 5, increase the final digit by 1

    Rounding to one decimal place: 5.798 → 5.8
  - **Rule 3**: If the remainder is exactly 5 then round the last digit to the closest even number

    This is to prevent rounding bias. Remainders from 1 to 5 are rounded down half the time and remainders from 6 to 10 are rounded up the other half.

    Rounding to one decimal place: 3.55 → 3.6, also 3.65 → 3.6

Examples

Example The period of a pendulum is given by $T = 2\pi\sqrt{l/g}$

Here, $l = 0.24\text{m}$ is the pendulum length and $g = 9.81\text{m/s}^2$ is the acceleration due to gravity

- ✘ **WRONG**: $T = 0.98326923\ 5922\ \text{s}$
- ✓ **RIGHT**: $T = 0.98\,\text{s}$

Note You can obtain the first number by a calculator but there is no way you know *T* to that level of precision

When no uncertainties are given, report your value with the same number of significant digits as the value with the smallest number of significant digits

Examples

Example The mass of an object was found to be 3.56g with an uncertainty of 0.032g

- ✘ **WRONG**: $m = 3.56 \pm 0.032\ \text{g}$
- ✓ **RIGHT**: $m = 3.56 \pm 0.03\ \text{g}$

Note The first way is wrong because the uncertainty should be reported with one significant digit

Example The length of an object was found to be $2.593\,\text{cm}$ with an uncertainty of $0.03\,\text{cm}$

- ✘ **WRONG**: $L = 2.593 \pm 0.03\text{cm}$
- ✓ **RIGHT**: $L = 2.59 \pm 0.03\text{cm}$

Note The first way is wrong because it is impossible for the third decimal point to be meaningful

Examples

Example The velocity was found to be $2.45\,\mathrm{m/s}$ with an uncertainty of $0.6\,\mathrm{m/s}$

- ✗ **WRONG**: $v = 2.5 \pm 0.6\,\mathrm{m/s}$
- ✓ **RIGHT**: $v = 2.4 \pm 0.6\,\mathrm{m/s}$

Note The first way is wrong because the first discarded digit is a 5 In this case, the final digit is rounded to the closest even number, 4

Example The distance was found to be $45600\,\mathrm{m}$ with an uncertainty of $1\mathrm{m}$

- ✗ **WRONG**: $d = 45600\,\mathrm{m}$
- ✓ **RIGHT**: $d = 4.5600\mathrm{x}10^{4}\,\mathrm{m}$

Note The first way is wrong because it tells us nothing about the uncertainty. Scientific notation shows we know the value to within 1m

Statistical Analysis of Small Data Sets

Repeated measurements allow you:

- To obtain a better idea of the actual value
- To characterize the uncertainty of your measurement

There are a number of quantities that are very useful in data analysis, that exploits:

- The value obtained from a particular measurement, x
- The times a measurement is repeated, N

Statistical Analysis of Small Data Sets

Oftentimes (e.g. in lab) *N* is small, usually no more than 5 to 10

For small data sets we use the following formulas:

- **Mean - x_{avg}** The average of all values of x (the "best" value of x)

$$x_{\text{avg}} = \frac{x_1 + x_2 + \cdots + x_N}{N}$$

- **Range - R** The "spread" of the data set. This is the difference between the maximum and minimum values of x

$$R = x_{\text{max}} - x_{\text{min}}$$

Statistical Analysis of Small Data Sets

- **Uncertainty in a measurement - Δx**

Determine this uncertainty by making multiple measurements.

From data you know that x lies somewhere between x_{max} and x_{min}

$$\Delta x = \frac{R}{2} = \frac{x_{\text{max}} - x_{\text{min}}}{2}$$

- **Uncertainty in the Mean - $\Delta\, x_{\text{avg}}$**

The actual value of x will be somewhere in a neighborhood around x_{avg}.

This neighborhood of values is the uncertainty in the mean.

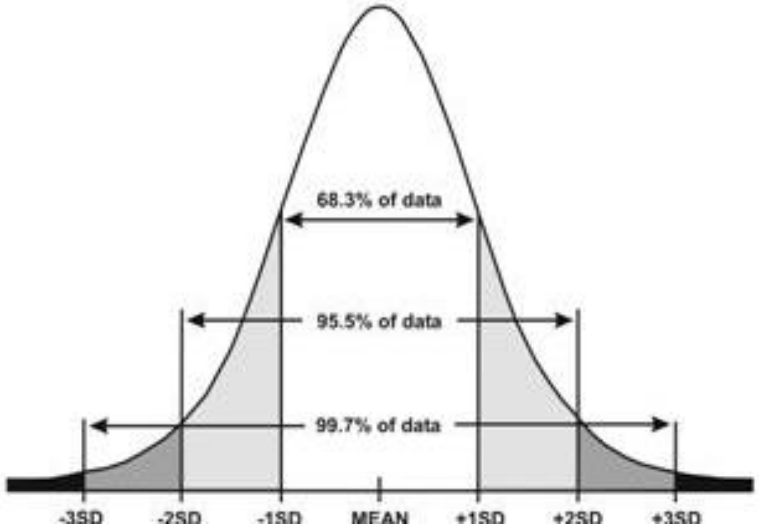
$$\Delta x_{\text{avg}} = \frac{\Delta x}{\sqrt{N}} = \frac{R}{2\sqrt{N}}$$

- **Measured value - x_{m}** The final reported value of a measurement of x contains both the average value and the uncertainty in the mean

$$x_{\text{m}} = x_{\text{avg}} \pm \Delta x_{\text{avg}}$$

Statistical Analysis of Large Data Sets

- If only random errors affect a measurement, it can be shown mathematically that in the limit of an **infinite** number of measurements ($N \to \infty$), the distribution of values follows a **normal distribution** (i.e. the bell curve)

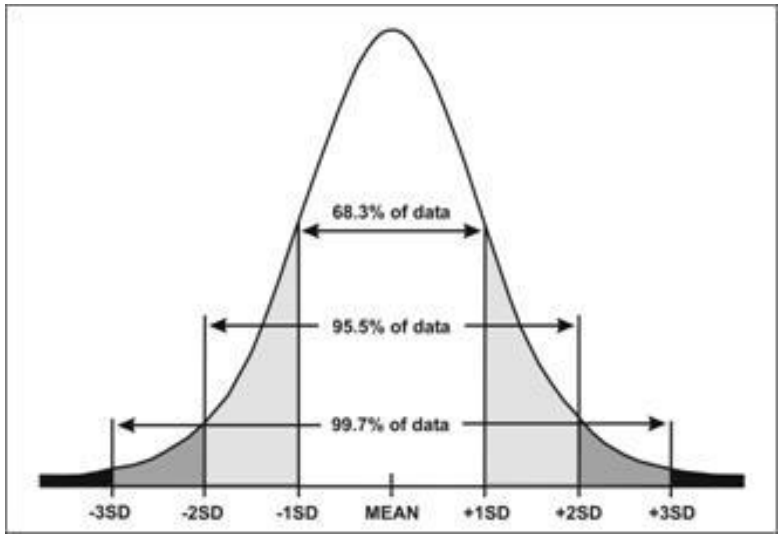


- This distribution has a **peak at the mean value** x_{avg} and a **width given by the standard deviation** σ

Statistical Analysis of Large Data Sets

- We never take an infinite number of measurements

- However, for a large number of measurements, N~10-10^2 or more, measurements may be approximately normally distributed



Statistical Analysis of Large Data Sets

For large data sets we use the following formulas:

- **Mean - x_{avg}** The average of all values of x (the "best" value of x)
  This is the same as for small data sets

$$x_{\text{avg}} = \frac{\sum_{i=1}^{N} x_i}{N}$$

- **Uncertainty in a measurement - Δx** The vast majority of your data lies in the range $x_{\text{avg}} \pm \sigma$

$$\Delta x = \sigma = \sqrt{\frac{\sum_{i=1}^{N} (x_i - x_{avg})^2}{N}}$$

Statistical Analysis of Large Data Sets

For large data sets we use the following formulas:

- **Uncertainty in the Mean - Δx_{avg}** The actual value of x will be somewhere in a neighborhood around x_{avg}. This neighborhood of values is the uncertainty in the mean

$$\Delta x_{avg} = \frac{\sigma}{\sqrt{N}}$$

- **Measured Value - x_m** The final reported value of a measurement of x contains both the average value and the uncertainty in the mean

$$x_m = x_{avg} \pm \Delta x_{avg}$$

Propagation of Uncertainties

- Oftentimes we combine multiple values, each of which has an uncertainty, into a single equation
- The way these uncertainties combine depends on how the measured quantity is related to each value
- Rules for how uncertainties propagate are:

- Addition/Subtraction $z = x + y$ $\Delta z = \sqrt{(\Delta x)^2 + (\Delta y)^2}$
- Multiplication $z = xy$ $\Delta z = |xy| \sqrt{\left(\frac{\Delta x}{x}\right)^2 + \left(\frac{\Delta y}{y}\right)^2}$
- Division $z = xy$ $\Delta z = \left|\frac{x}{y}\right| \sqrt{\left(\frac{\Delta x}{x}\right)^2 + \left(\frac{\Delta y}{y}\right)^2}$

Examples

- **Addition** The sides of a fence are measured with a tape measure to be 124.2cm, 222.5cm, 151.1cm and 164.2cm
- Each measurement has an uncertainty of 0.07cm
- Calculate the measured perimeter P_{m} including its uncertainty

$$P = 124.2\mathrm{cm} + 222.5\mathrm{cm} + 151.1\mathrm{cm} + 164.2\mathrm{cm} = 662.0\mathrm{cm}$$

$$\Delta P = \sqrt{(0.07)^2 + (0.07)^2 + (0.07)^2 + (0.07)^2} = 0.14\mathrm{cm}$$

$$P_{\mathrm{m}} = 662.0 \pm 0.1\mathrm{cm}$$

Examples

- **Multiplication** The sides of a rectangle are measured to be 15.3cm and 9.6cm
- Each length has an uncertainty of 0.07cm
- Calculate the measured area A_{m} including its uncertainty

$$A = 15.3\mathrm{cm} \times 9.6\mathrm{cm} = 146.88\mathrm{cm}^2$$

$$\Delta A = 15.3\mathrm{cm} \times 9.6\mathrm{cm} \sqrt{\left(\frac{0.07}{15.3}\right)^2 + \left(\frac{0.07}{9.6}\right)^2} = 1.3\mathrm{cm}^2$$

$$A_{\mathrm{m}} = 147 \pm 1\,\mathrm{cm}^2$$