Introduction
to
Computational Science

PART 1 OF 4

Overview
Foundations of Modeling
Tools and Techniques for Modeling - Mathematica

James L. Noyes, Ph.D.
Professor Emeritus, Wittenberg University

January 2011
Overview
(See [Gior2009; pp. xiii-xvi, p. 1])

Introduction, Prerequisites and Other Details

The Process of Science

The Philosophy of Science

Foundations for Computational Science
- Models
- Errors
- Physical Units
- Dimensional Analysis
- Tools
- Using Mathematica Command Functions
- Programming (Coding) in Mathematica

Scientific Models and Methods for their Solution
- Model Categories
- Evaluation Models
- Simulation Models
- Optimization Models
Introduction

Computational Science: A scientific investigation through the modeling, simulation, and analysis of physical processes on a computer. Hypotheses are often formulated using mathematical models that can then be used to compute values of scientific interest and gain new insights.

Computational science is now considered by most scientists to be on par with the development of scientific theory and the use of experimentation in order to understand more about our world.

Computational science is not the same as computer science. Rather, it is an interdisciplinary blend of scientific models, applied mathematics, computational techniques, and practices.

COSC = Science Problem + Math Model + Computational Method

Computational science is using the computer and mathematics to help understand science through the use of numeric, symbolic, and graphics processing.

This Introduction to Computational Science course focuses upon simple and intuitive computational models and methods.
Prerequisites

1. Fundamental **Computing Experience**: The basic understanding of how a (sequential) computer works and the key constructs involved in programming (e.g., as found in programming languages such as C, C++, Java, FORTRAN, Lisp, etc.). This includes fundamental topics such as: algorithms, data types, data storage, I/O, loops, branches, subprograms, files, arrays, and documentation. However, one does NOT need to have a lot of coding experience. In this course the student will be taught enough *Mathematica* to be able to state and solve problems when the models and algorithms to solve them are given. It means enough experience to effectively implement given algorithms and data structures necessary in solving scientific problems. COMP 150 more than satisfies this.

2. Fundamental **Mathematical Ability**: The basic capability to understand and use both discrete and continuous mathematical models based upon algebra, elementary functions, geometry, and differential calculus. It means enough ability to be capable of solving problems by using these models and the ability to learn how to use vectors, matrices, infinite sequences, series, finite differences, integrals, regular and partial derivatives in solution algorithms. MATH 131 or MATH 201 will satisfy this.

The REAL Prerequisite – **Desire to Learn and Explore New Ideas**: To seek out needed information (through studying, computational experimentation, and asking questions in and out of class).
Course Content, Goals, and Requirements

Course Content: The syllabus is given as a handout. However, each student should check out other related computational science links on the Web and the provided references at the end of the course notes. This will be a breadth-oriented survey course, covering a lot of topics, rather than a few topics done in depth.

Course Goals:
- Understand the Scientific Process and the Philosophy of Science
- Understand the Purpose and Value of Computational Science
- Learn to Use Computer-Aided Problem Solving and Visualization Techniques for Science-Related Problems. *Mathematica* is Used
- Acquire the Skill to Use and Modify Scientific Models. Understand the Assumptions Involved.
- Acquire the Skill to Select the Methods Needed to Solve Scientific Models and to Know their Strengths and Limitations.

Course Requirements:
- Get Involved. Actively Participate in Class with both Questions and Answers
- Think About What is Important. Anticipate. Think Science!
- Independently Read and Experiment with *Mathematica* Examples.
- Record and Organize Work Periodically (Keep a Notebook)
- Do Homework Assignments.
Potential Topics

PART 1

Overview
- Introduction
- Prerequisites
- Course Content, Goals, Requirements
- Potential Topics

Foundations of Modeling
- The Study of Science
- The Scientific Process
- Scientific Research
- Philosophy of Science
- Science and Ethical Values
- Introduction to Computational Science
- The Problem Solving Paradigm of Computational Science
- Simplified Example of Problem Solving Paradigm
- Problem Solving Revisited - What Can Go Wrong
- Model Space: Classes of Mathematical Models
- Analytic vs. Numerical Solution Methods
- Physical Units, Constants, and Conversion Factors
- SI Units
- SI Prefixes and Usage
- Dimensional Analysis
- Sources of Error
- Absolute, Relative and Percent Error
- Approximation Error

Tools and Techniques for Modeling - Mathematica
- Hardware and Software Tools for Computational Science
- Mathematica®
- Capabilities of the Mathematica System
- Interpreted vs. Compiled Programming Languages
- Using the Mathematica Interpreter
- Some Useful Mathematica Kernel Functions
- Numerical Computation with Mathematica

PART 2

Discrete Modeling and Computational Considerations
- Sequences and Series
• Difference Equations and Operators
• State Transition Diagrams
• Solving Models of Difference Equations
• Basic Recurrence Relations
• Real vs. Floating-Point Number Systems
• Floating-Point and Integer Arithmetic
• Underflow, Overflow, Overflow and Roundoff Errors
• Integer and Floating-Point Computation
• Sensitivity of Floating-Point Operations
• Conditioning of a Problem
• Rules for Reducing Numerical Errors
• Types of Scientific Models
• Methods for Problem Analysis and Solution
• Performing a Case Study in Problem Solving
• Interval Arithmetic

Developing Continuous Functional Approximations and Models
• Functions and Mathematical Models
• Defining and Using Functions in Mathematica
• The Taylor Series and the Remainder Term
• Approximating Functions and Data
• The Lagrange Polynomial and the Error Term
• Multivariate Functions, Vectors, and Derivatives
• Least-Squares Approximation
• Least-Squares Polynomials
• Linear Combinations of Basis Functions
• Nonlinear Models That Can Be Linearized
• Characteristics of Models
• Least-Squares Approximation with Mathematica
• Interpolation vs. Least-Squares in Mathematica
• Observation and Scientific Laws
• The Experimental Method and Scientific Discovery
• Multivariable Scientific Relationships
• Sensitivity of Models

PART 3

Linear System and Differential Equation Modeling
• Scalars, Vectors, and Matrices
• Vector and Matrix Computations
• Solving Linear and Matrix Systems
• Matrix Eigenvalues and Eigenvectors
• Vector and Matrix Norms
• Ill-Conditioned Matrices
• Vector and Matrix Operations in Mathematica
• Analytic Solution of Difference Equations
• Scientific Visualization
• Visualization with Mathematica
• Difference Equations vs. Differential Equations
• ODE/IVP Discretization: The Euler Method
• ODE Implementation in Mathematica
• Solving Differential Equations with Mathematica: Symbolic and Numeric
• Integration with Mathematica: The Trapezoidal Rule
• ODE Systems and $N^{th}$ Order ODEs
• Partial and Integral Equation Models

Stochastic Modeling
• Random and Pseudo-Random Numbers
• Full Period PRN Generator
• Pseudo-Random Numbers and Simulation
• Generating Uniform Pseudo-Random Numbers
• Using Uniform PRNs in Mathematica
• Other Statistical Distributions
• Using Selected Distributions in Mathematica
• Stochastic Simulation with Mathematica

Optimization Modeling
• Roots/Zeros of Nonlinear Equations
• Bisection Method
• Newton Method
• Writing a Bisection Function in Mathematica
• Equation Solving and Root-Finding in Mathematica
• Newton’s Method for Nonlinear Systems
• Simple Mathematica I/O: Example with a File
• Optimization: Minimization or Maximization (Unconstrained and Constrained)
• Unconstrained Optimization Methods
• Unconstrained Optimization in Mathematica
• Classical Optimization with Equality Constraints: The Lagrange Method
• Optimality Conditions for Unconstrained Minimization Problems
• Constrained Optimization with Mathematica: Linear and Nonlinear Programming

PART 4

References
Foundations of Modeling
(See [Gior2009; pp. xviii])
The Study of Science

As defined by Arthur Strahler [LeeJ2000], “Science is the acquisition of reliable but not infallible knowledge of the real world, including explanations of the phenomena.” The realm of science can involve the empirical investigation and analysis of anything that is observable in our known universe.

The major areas of science are commonly grouped into the natural and behavioral sciences.

The natural sciences include the physical sciences, biological sciences, and earth sciences. The physical sciences, physics and chemistry, deal with matter and energy. The biological sciences, such as biology, botany, and zoology, deal with living organisms and their interactions. The earth sciences, such as geology, deal with nonliving matter.

The behavioral sciences deal with the behavior of individuals, through psychology, and groups, through the social sciences.

Some scientific disciplines cross boundaries of these categories. These include astronomy (earth and physical sciences), biochemistry (biological and physical sciences), neurology (biological, behavioral, and physical sciences). Interdisciplinary studies are becoming much more common in science.
The Scientific Process

The scientific process starts by *observing* scientific data and *thinking* about what it means (e.g., cause and effect).

- Form Hypothesis (make predictions based on this)
- Experimentation/Observation (collect and analyze data)
- Theory Confirmation/Refutation (support or reject)

The goal of science is to gain a better understanding about physical scientific processes (the “laws that *describe* nature”). A good hypothesis should have a strong explanatory power and must be one that can be disproved.

**Computational Science**: A scientific investigation through the modeling, simulation, and analysis of scientific processes on a computer. Hypotheses are often formulated using mathematical models that can then be used to compute values of scientific interest and gain new insights.

**Role of Communication** – one must be able to read and interpret the works of others and clearly explain and persuade others of the value of worthy new ideas.

**Roles of Philosophy and Ethics** – one should be aware of how a specialized scientific discipline relates to other important disciplines, how it fits into the whole of society, and how it should be ethically used.
Scientific Research

1. The commonly employed “positivist” approach to scientific research involves first selecting a topic of interest or importance and one in which there is a sufficient amount of technical expertise and available resources.

2. Based upon careful examination and measurement of relevant data, a hypothesis is formulated that may explain a particular phenomenon. This hypothesis is typically based upon analogy, deduction, induction, or intuition, and should satisfy the following criteria [LeeJ2000]:
   • It should adequately explain the specified phenomenon
   • It should be internally consistent
   • It should be consistent with most available knowledge
   • It should have the potential of improving the existing theory
   • It should make predictions which are empirically testable

   Often this hypothesis utilizes a mathematical model (as a hypothesis generator), which attempts to identify constants, parameters, independent and dependent variables. The model’s strength and limitations must be understood.

3. The hypothesis is tested, based upon a careful and thorough unbiased experimental design plan. Direct observations (preferred) or indirect observations (when necessary) are made and carefully documented.

4. The experimental information is carefully analyzed to determine if the hypothesis is sufficiently supported. If not, it has to be modified (and re-tested) or discarded.
Philosophy of Science

According to one of the best-known philosopher’s of science, Karl Popper, science is a dichotomy. On one hand, we know a great deal about both the theoretical and practical aspects of science. On the other, there are almost endless things that we do not know or only think we know, due to limited sensory experience [Popp1968, LeeJ2000].

Philosophy helps those that study science, gain an understanding of scientific phenomena as well as understand the limitations of the knowledge that is currently possessed.

As in all areas of philosophy, there are differing views as to exactly what constitutes the philosophy of science. One popular view is that it is a discipline in which the concepts and the theories of the sciences are investigated and their exact meaning is explained. In addition, the philosopher of science attempts to address the following questions [Lose1993]:

- What are the characteristics that distinguish scientific inquiry from other types of investigation?
- What are the procedures that scientists should follow in order to investigate phenomena?
- What are the conditions that must be satisfied in order for a scientific explanation to be correct?
• What is the cognitive status of scientific laws and principles?

In short, the philosophy of science deals with the assumptions, procedures, and explanations that scientists employ, in order to give them a stronger foundation. This study of science goes back to Aristotle (384BC - 322BC), the Grecian scientist-philosopher who studied under Plato.

Scientific statements try to describe real phenomena and scientific knowledge should represent the most accurate view of the real world that is possible. For a statement to be truly scientific, it must be *testable*, and tests must be *reproducible* by others.

The so-called scientific method is really an organized collection of approaches designed to acquire scientific knowledge. Here empiricism is combined with logical reason.

In *induction*, one reasons from the particular to the general. A scientist measures a phenomenon, analyzes the measurements, and proposes a theory (a hypothesis) that explains and predicts. In *deduction*, one starts with a theory, collects and measures relevant data, and analyzes it. Deduction is thought to be more efficient and reliable. Mathematics ties this all together as shown below:

**Mathematics:** Theories $\rightarrow$ Deduction $\rightarrow$ Predictions

$\begin{array}{c}
\text{Reality:} \\
\text{Facts} \uparrow(\text{Induction}) \\
\text{Facts} \downarrow(\text{Verify})
\end{array}$
Arguably the best-known scientific approach is that of the previously stated positivist approach [Keme1959].

A modification of this positivist approach involves what Karl Popper called falsification. Here the scientist makes a hypothesis and tries to show that it is not correct. If a case makes it fail, the hypothesis can be rejected or again modified (after verifying that the “counterexample” case is itself legitimate). Further experimental cases are designed to falsify the modified version. When the hypothesis has survived many strong tests, it is considered to be a reliable theory. If most scientists who specialize in this particular scientific discipline are convinced that the hypothesis will pass every conceivable test, the theory is “upgraded” into a law. Note: Despite being called a law it is NOT PROVED.

Establishing a connection between mathematics and reality is one of the most difficult tasks for a scientist. This involves the development of a sufficiently accurate mathematical model. The most valuable models express quantitative relationships among its variables. This enables it to have a quantitative predictive power.

The facts are known and particular, since they refer to a single event or a series of events. However, theories are universal in nature, and therefore can never be known to be entirely true. It is important that these observations are made under circumstances that insure that no bias is introduced.
Example: Suppose one might observe values of \{0, 20\pi, 40\pi, \ldots\} at the time points \{0, 2\pi, 4\pi, \ldots\}. One might infer that \(f(x) = 10x\) is a reliable the model. However, additional observations at \{\pi/4, \pi/2, 3\pi/4, \pi, \text{ etc.}\} would cause a major model revision to be given as \(g(x) = 10 \cdot x \cdot \cos(x)\). Here what first appeared to be obvious, the model \(f(x)\), turned out to be wrong. The model \(f(x)\) was incorrectly concluded because not enough data was collected and analyzed. Based upon the data, a better model is \(g(x)\).

Example: Newton’s theories were based upon the mass of an object being a constant \(m\) (now called a rest mass). Einstein showed that the mass of an object depends upon the velocity, \(v\), that it has, namely \(M = m / \sqrt{1 - (v/c)^2}\). Here \(c\) is the speed of light. Newton’s theory is a good approximation until the velocity of the object approaches the speed of light. Here Newton’s theory is not as universal as first supposed.

Perceptions are mental images of the external environment. These are based upon input from the senses and the memory of previous experience. Due to the related subjectivity, it is important that factual observations be taken with quantitative sensory devices. In addition, these observations must be repeated, often by different observers, in order to remove any measurement errors or subjective biases in interpretation. However, even sensory devices cannot always give a complete and accurate picture of “objective reality.” Sometimes a statistical theory is necessary.
*Induction* in science means deriving a general statement (hypothesis, theory, or law) from a set of repeated statements or observations that are all in agreement. This general statement is sometimes called a *universal statement*. This statement deals with a *regularity of nature*. Unfortunately, it IS NOT possible to *prove* a universal statement from a set of observations. It IS, of course, possible to *disprove* a universal statement based upon just one observation. For it to be worthwhile, this universal statement must have predictive power and be specific enough to be disproved.

*Deduction* in science starts by assuming the general or universal statement is true. It then typically uses logic and often mathematics to predict what should be true under a given set of conditions.

There are several varieties of scientific theories and laws. These have different forms and uses such as to describe, to explain, and to predict. Here are four types:

- Property laws state that a particular object or substance always possesses a specified property. These tend to be rather specific.
- Causal laws tell how a particular cause is always followed by a specific effect.
- Statistical laws concern the probability of a certain event occurring in a series of events.
- Functional laws provide a functional dependence among variables that is associated with certain properties or processes.
One more time … Laws of science are NOT absolute truths (even though we frequently treat them that way), because there is no absolute certainty that a law that has always applied in the past will apply through all future times or that it applies in all other realms of space other than that in which it has been verified [Stra1992].
Science and Ethical Values

Science is not an activity that is free of ethical values since it is influenced by cultural, economic, and political influences. However some areas of science (e.g., biology) have more outside influence than other areas.

- The directions of scientific research are influenced by the goals of the institutions where the scientists work.

- Scientists have personal motivations that are based upon previous experience. Usually the scientist cannot directly report this in their findings, but this may implicitly affect their work.

- Because of the extraordinary success of science in solving important problems, scientists are often assumed to be infallible and unbiased. There is a temptation to neglect the limitations of science.

- In the past, religious authorities have attempted to dictate scientific conclusions. Now many treat science almost as a religion, by having complete faith in scientific pronouncements. There are still many unexplained events in the universe.
Introduction to Computational Science

• Reasons for Study:

To help understand how scientific models help describe and explain what happens in the world. Specifically, this means to:

1. Carefully analyze the problem.
2. Choose the best model to apply.
3. Determine when this model is valid (what assumptions are necessary).
4. Select or develop appropriate (approximate) solution methods.
5. Implement these methods effectively on the computer (using numeric, symbolic, and graphical processing).
6. Verify for what situations these computational methods are correct.
7. Validate the final solution with respect to the original problem.

• Postulation, Theory, Experiment, and Computation:

Computational science is now considered to be a new paradigm of scientific investigation (in addition to scientific postulation, theory, and experiment) and often yielding new insights [Glim1987].
Problem Solving Paradigm of Computational Science

- Identify the Problem, Major Variables and Relevant Data
- Select or Build a Representative Model
- Validate the Model with Respect to the Problem
- Select or Develop the Necessary Solution Methods
- Effectively Implement the Solution Methods in Software
- Execute the Computational Method & Verify its Results
- Assess the Results with Respect to the Original Problem
- Optionally Modify the Appropriate Steps and Repeat
- Describe the Results and their Importance
Simplified Example of Problem Solving Paradigm

- Identify the Problem, Major Variables and Relevant Data

For the given (time, temperature) data (in minutes and Fahrenheit degrees), estimate the time when the temperature reaches a given value. Specifically for \{(0, 0.01), (2, 9.48), (7, 33.27), (8, 37.97), (9, 42.76)\}, temp=20.

- Select or Build a Representative Model

Based upon examination of the data, the model \(F(t) = at\) looks like a reasonable one, where \(F\) is the Fahrenheit temperature and \(t\) is the time in minutes. The parameter \(a\) must be found so that this model will best “fit” the data. A common method to do this is to minimize the sum of squares of the vertical errors. That is, Minimize \(E(a) = \frac{1}{2} \sum [at_i - F_i]^2\) with respect to the parameter \(a\), summed over \(n = 5\) points.

To solve for \(a\), one finds where \(\frac{dE(a)}{dt} = 0\). This yields the parameter \(a = \frac{\sum(t_iF_i)}{\sum t_i^2}\). This yields \(a = 940.45/198 \approx 4.750\). The model is: \(F = 4.75t\).

- Validate the Model with Respect to the Problem

\[E = \frac{1}{2}[(0-0.01)^2 + (9.50-9.48)^2 + (33.25-33.27)^2 + (38.00-37.97)^2 + (42.75-42.76)^2] \approx 9.5 \times 10^{-4}\]

- Select or Develop the Necessary Solution Methods

Here all one needs to do is to solve: \(t = F/4.75\) for a given value of \(F\).
• Effectively Implement the Solution Methods in Software

Implement: Input $F$
Compute $t = F/4.75$
Output $t$

• Execute the Computational Method & Verify its Results

$t \approx 20/4.75 \approx 4.21$ minutes

• Assess the Results with Respect to the Original Problem

This result is reasonable one since this value is between $t = 2$ and $t = 7$. Graphing would give a better insight. Would there be any difficulty in solving this problem if either $F = -9$ or $F = 75$?

• Optionally Modify the Appropriate Steps and Repeat

• Describe the Results and their Importance
Problem Solving Revisited – What Can Go Wrong

Question: “What can go wrong?”

Answer: “Everything!”

Some things that can go wrong:

- Incorrect Understanding and Analysis of Problem
- Inexact Measurements (loss of significant digits)
- Inappropriate or Invalid Model (incorrect assumptions)
- Poor Choice of Method (incorrect or inefficient)
- Poor Implementation of Method (flawed computation)
- Poor Assessment of Results (erroneous validation)
- Inappropriate Discussion of Results and Confidence

Moral: “Unless you have thoroughly checked every step of the process, you must assume that your results contain errors.”

Comment: “No one said that this would be easy (important, but not easy)!}
Model Space: Classes of Mathematical Models

There are various ways to organize models. The following is one of the most useful for scientific modeling purposes. It is based upon three independent factors (three axes):

- **Model Continuity**: *Continuous* vs. *Discrete*
  variables/parameters are continuous or can have discrete values

- **Model Time-Dependence**: *Static* vs. *Dynamic*
  variables/parameters are not or are functions of time

- **Model Knowledge**: *Deterministic* vs. *Stochastic*
  variables/parameters are completely determined or random

This yields a total of eight “pure” model categories, with hybrid models that have some of more than one factor. This can be used to develop a model taxonomy that will be arbitrarily numbered below. Each can be *linear* or *nonlinear*.

Category 1 (CSD): Continuous, Static, and Deterministic
Category 2 (CDD): Continuous, Dynamic, and Deterministic
Category 3 (DSD): Discrete, Static, and Deterministic
Category 4 (DDD): Discrete, Dynamic, and Deterministic
Category 5 (CSS): Continuous, Static, and Stochastic
Category 6 (CDS): Continuous, Dynamic, and Stochastic
Category 7 (DSS): Discrete, Static, and Stochastic
Category 8 (DDS): Discrete, Dynamic, and Stochastic

Each of these models may be used for *evaluation*, *simulation*, or *optimization* purposes.
Analytic vs. Numerical Solution Methods

The primary advantage of analytic so-called “closed form” solution methods is that one may examine this form (i.e., formulas and equations) and investigate the behavior by changing parameters and variables and observing the consequences. Analytic solutions are compact in that a relatively small amount of very descriptive information is needed. Whenever possible, analytic solution methods should be used in preference to numeric methods. Note that the solutions obtained by these methods are usually evaluated numerically and investigated graphically. Unfortunately, it may be very difficult or impossible to find analytic solutions to the more realistic models.

The primary advantage of numerical solution methods is that they can almost always be used, even for the most difficult models. These methods typically produce a few values (numbers) or perhaps an array of values that may be examined and easily graphed. However, if a parametric study is required, the entire problem must be re-solved for each parameter value desired. In addition, numerical methods are often subject to several types of computational errors including roundoff error, accumulation error, truncation error, and the like.
Physical Units, Constants, and Conversion Factors

Most physical quantities can be expressed in terms of combinations of seven basic dimensions: **mass** (M), **length** (L), **time** (T), **electric current** (I), **temperature** (Γ), **luminous intensity** (Y), and **amount of substance** (A). These dimensions are not the same as specific units, which depend upon one or more unit systems such as the SI (Systeme International) System and the English System. From these basic dimensions, others may be derived such as **energy** (M·L²/T²), **force** (M·L/T²), **velocity** (L/T), **acceleration** (L/T²), **power** (M·L²/T³), and **charge** (I·T).

Some quantities, such as those defined by ratios of equivalent dimensions are themselves dimensionless. For example, the sine of an angle is L/L, and so is dimensionless. All trigonometric functions, exponential functions, and logarithms are dimensionless, as are angles and other arguments. For example, if one has e^{kt}, where t is of dimension T, then k must be of dimension T⁻¹. Some dimensionless quantities have units (e.g., angles in radians).

Physical units, such as SI units, help specify modeling formulas in a common reference system. A formula must use a **consistent** set of units. An incorrect unit is just as wrong as an incorrect value. Within a given system, physical constants and conversion factors (between and among systems) may be defined.
# SI Units

## SI Base Units:

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Name</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>length</td>
<td>meter</td>
<td>m</td>
</tr>
<tr>
<td>mass</td>
<td>kilogram</td>
<td>kg</td>
</tr>
<tr>
<td>time</td>
<td>second</td>
<td>s</td>
</tr>
<tr>
<td>temperature</td>
<td>kelvin</td>
<td>K</td>
</tr>
<tr>
<td>electric current</td>
<td>ampere</td>
<td>A</td>
</tr>
<tr>
<td>amount of substance</td>
<td>mole</td>
<td>mol</td>
</tr>
<tr>
<td>luminous intensity</td>
<td>candela</td>
<td>cd</td>
</tr>
</tbody>
</table>

## Selected SI Derived Units:

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Name (Symbol)</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>area</td>
<td>meter squared m²</td>
<td>m²</td>
</tr>
<tr>
<td>volume</td>
<td>meter cubed m³</td>
<td>m³</td>
</tr>
<tr>
<td>velocity</td>
<td>meter per second m s⁻¹</td>
<td>m s⁻¹</td>
</tr>
<tr>
<td>acceleration</td>
<td>meter per second squared m s⁻²</td>
<td>m s⁻²</td>
</tr>
<tr>
<td>density</td>
<td>kilogram per meter cubed kg m⁻³</td>
<td>kg m⁻³</td>
</tr>
<tr>
<td>plane angle</td>
<td>radian (rad)</td>
<td>rad</td>
</tr>
<tr>
<td>angular velocity</td>
<td>radian per second rad s⁻¹</td>
<td>rad s⁻¹</td>
</tr>
<tr>
<td>frequency</td>
<td>hertz (Hz)</td>
<td>s⁻¹</td>
</tr>
<tr>
<td>force</td>
<td>newton (N)</td>
<td>m kg s⁻²</td>
</tr>
<tr>
<td>pressure (stress)</td>
<td>pascal (Pa)</td>
<td>N m⁻²</td>
</tr>
<tr>
<td>energy (work)</td>
<td>joule (J)</td>
<td>N m</td>
</tr>
<tr>
<td>power</td>
<td>watt (W)</td>
<td>J s⁻¹</td>
</tr>
<tr>
<td>electric charge</td>
<td>coulomb (C)</td>
<td>A s</td>
</tr>
<tr>
<td>electric potential</td>
<td>volt (V)</td>
<td>W A⁻¹</td>
</tr>
<tr>
<td>electric resistance</td>
<td>ohm(Ω)</td>
<td>V A⁻¹</td>
</tr>
</tbody>
</table>

* The radian is m m⁻¹ = 1, and is a dimensionless derived unit.
## SI Prefixes and Usage

<table>
<thead>
<tr>
<th>Name</th>
<th>Factor</th>
<th>Abbreviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>yocto (septillionth)</td>
<td>$10^{-24}$</td>
<td>y</td>
</tr>
<tr>
<td>zepto (sextillionth)</td>
<td>$10^{-21}$</td>
<td>z</td>
</tr>
<tr>
<td>atto (quintillionth)</td>
<td>$10^{-18}$</td>
<td>a</td>
</tr>
<tr>
<td>femto (quadrillionth)</td>
<td>$10^{-15}$</td>
<td>f</td>
</tr>
<tr>
<td>pico (trillionth)</td>
<td>$10^{-12}$</td>
<td>p</td>
</tr>
<tr>
<td>nano (billionth)</td>
<td>$10^{-9}$</td>
<td>n</td>
</tr>
<tr>
<td>micro (millionth)</td>
<td>$10^{-6}$</td>
<td>μ</td>
</tr>
<tr>
<td>milli (thousandth)</td>
<td>$10^{-3}$</td>
<td>m</td>
</tr>
<tr>
<td>centi (hundredth)</td>
<td>$10^{-2}$</td>
<td>c</td>
</tr>
<tr>
<td>deci (tenths)</td>
<td>$10^{-1}$</td>
<td>d</td>
</tr>
<tr>
<td>deka (ten)</td>
<td>$10^{1}$</td>
<td>da</td>
</tr>
<tr>
<td>hecto (hundred)</td>
<td>$10^{2}$</td>
<td>h</td>
</tr>
<tr>
<td>kilo (thousand)</td>
<td>$10^{3}$</td>
<td>k</td>
</tr>
<tr>
<td>mega (million)</td>
<td>$10^{6}$</td>
<td>M</td>
</tr>
<tr>
<td>giga (billion)</td>
<td>$10^{9}$</td>
<td>G</td>
</tr>
<tr>
<td>tera (trillion)</td>
<td>$10^{12}$</td>
<td>T</td>
</tr>
<tr>
<td>peta (quadrillion)</td>
<td>$10^{15}$</td>
<td>P</td>
</tr>
<tr>
<td>exa (quintillion)</td>
<td>$10^{18}$</td>
<td>E</td>
</tr>
<tr>
<td>zetta (sextillion)</td>
<td>$10^{21}$</td>
<td>Z</td>
</tr>
<tr>
<td>yotta (septillion)</td>
<td>$10^{24}$</td>
<td>Y</td>
</tr>
</tbody>
</table>

Usage: All unit symbols are written with lowercase letters unless the unit is named after a person. Plural is never used. Multiplication of units is indicated by a blank space between or a dot (\(\cdot\)). A period is never used (except at the end of a sentence). Division of units is indicated by a negative exponent, horizontal line, or forward slash (/). A blank space is normally used between the number and unit.
Dimensional Analysis

**Dimensional analysis**, as proposed by P.W. Bridgman, is a problem solving methodology whereby units are treated as algebraic quantities so that quantitative relationships may be converted, simplified, and verified. The goal is often to cancel out all unwanted units and leave only the desired ones. There are a few basic rules of dimensional analysis:

- In an expression, all terms which are added or subtracted must have the same dimension.
- The left and right side of an equation must have the same dimension.
- Quantities of different dimensions may be multiplied or divided, but not added or subtracted.
- When multiplying or dividing by conversion factors, each conversion factor must be equal to unity.

**Conversion Examples:**

1. How many inches in 12 meters?
   \[12\text{m} \times \left(\frac{100\text{cm}}{\text{m}}\right) \times \left(\frac{1\text{in}}{2.54\text{cm}}\right) = 472.44\text{in}\]

2. Given that the speed of light \(c\) is 299,792,458m/s, a furlong is 220 yards, and a fortnight is 14 days, what is \(c\) in furlongs per fortnight?
   \[299792458\text{m/s} \times \left(\frac{100\text{in}}{2.54\text{m}}\right) \times \left(\frac{1\text{yd}}{36\text{in}}\right) \times \left(\frac{1\text{fur/220yd}}{}\right) \times \left(\frac{3600\text{s/hr}}{}\right) \times \left(\frac{24\text{hr/day}}{}\right) \times \left(\frac{14\text{day/ftn}}{}\right) = 1.8026175 \times 10^{12}\text{fur/ftn}\]

An equation or expression in which each term has the same dimensions is said to be dimensionally correct. This is a way to derive formulas and to check for algebraic errors.
Sources of Error

- **Adequacy of the Model**
  - Modeling Error
  - Model Sensitivity

- **Accuracy of the Data**
  - Data Error (Sensor Error)

- **Development of the Method**
  - Truncation Error
  - Approximation Error

- **Implementation of the Method**
  - Coding Blunders
  - Poor Code Formulation

- **Data Transcription**
  - Data Entry

- **Computation**
  - Roundoff
  - Underflow
  - Overflow
Absolute, Relative and Percent Error

Let \( q \) be an approximation to \( p \). The **absolute error** is

\[
| p - q |
\]

and is always defined. The **relative error** is

\[
\frac{|p - q|}{|p|}
\]

and is defined for \( p \neq 0 \). The **percent error** is

\[
100\frac{|p - q|}{|p|}
\]

for \( p \neq 0 \).

In practice, the true value \( p \) is most often unknown. What one usually has is the approximation \( q \), together with a value \( E \) that is an estimate of the error or a value \( B \) that is a bound for the error. Since it is not usually known if \( q \) is larger or smaller than \( p \), the absolute value is used.

**Example:** If \( B = 0.1 \) is a given bound on the absolute error for the approximation \( q = 6 \) to an unknown value \( p \), then one can say that \(|p - 6| \leq 0.1\), or \( 5.9 \leq p \leq 6.1 \).
Approximation Error

If the magnitude of the error in approximating $p$ (correct) by $q$ (approximation) does not exceed $0.5 \times 10^t$, that is

$$|p - q| \leq 0.5 \times 10^t.$$  

Then $q$ is said to be **correct to $t$ decimal places (rounded)**.

If the magnitude of the error in approximating $p$ by $q$ does not exceed $10^t$, that is

$$|p - q| \leq 10^t.$$  

Then $q$ is said to be **correct to $t$ decimal places (chopped)**.

The value $q$ is said to approximate $p$ to **$t$ significant digits (rounded)**, if $t$ is the largest non-negative integer for which

$$\frac{|p - q|}{|p|} \leq 0.5 \times 10^{t+1} = 5 \times 10^t.$$  

The value $q$ is said to approximate $p$ to **$t$ significant digits (chopped)**, if $t$ is the largest non-negative integer for which

$$\frac{|p - q|}{|p|} \leq 10^{t+1}.$$  

**Example:** Let $p = 123.456$, $q = 123.5$.  
Here $|p - q| = 0.044 \leq 0.5 \times 10^{-1}$, so $q = 123.5$ is correct to 1 decimal place (rounded).  
Also $|p - q|/|p| = 0.0003564022 \leq 5 \times 10^{-4}$, so $q = 123.5$ is correct to 4 significant digits (rounded).
Tools and Techniques for Modeling - Mathematica
Hardware and Software Tools
for
Computational Science

Calculator:
Programmable Scientific - Graphing Capability
Programmable Scientific - Graphing & Symbolic Capability

Computer (Sequential):
Some Parallel Co-processing
Compilers/Interpreters (C/C++, FORTRAN, Mathematica, MATLAB)
Spreadsheet Programs (Excel)
Math/Stat Utilities (BLAS, EISPAC, IMSL, LINPAC, SAS)
Specialized Math/Science Software (MINPAC, SLAM)
Visualization Software

Workstation/Mainframe:
Some Parallel Processes (RISC)
Compilers/Interpreters (C/C++, FORTRAN, Mathematica, MATLAB)
Math/Stat Utilities (BLAS, EISPAC, IMSL, LINPAC, SAS)
Specialized Math/Science Software (MINPAC, SLAM)
Visualization Software

Parallel Computer(s)/Supercomputer:
Significant Parallel Processing Capability (RISC)
Special Hardware Components
Compilers (C/CC++, FORTRAN)
Matrix/Vector Parallel Algorithm Libraries
Specialized Math/Science Software
Visualization Software
Mathematica®

Mathematica is an extremely flexible and powerful computer language that enables the user to perform numeric, symbolic, and graphical processing with a small amount of effort. It is available on many platforms. There are three ways for the user to interface with Mathematica - using its kernel, the notebook front end, or a package application.

Kernel: A large C program that deals with inputs and outputs by means of two processes: a) C code to perform various computations, and b) Rewrite-rules to reduce expressions into a normal form.

Notebook: The GUI front end is the most natural and common interface and is common among platforms. It provides facilities for editing and organizing text (such as this) and for sending inputs to the kernel for evaluation. The kernel sends results back to the front end, which is then responsible for displaying all of the numeric, symbolic, and graphical outputs in appropriate form. Documents developed in the notebook front end can be printed exactly as they appear on the screen.

Packages: Programs written in the Mathematica language that “extend” the language. Each package is designed to solve a more specific application or class of applications.
Capabilities of the Mathematica® System

Mathematica may be used at several levels [Gray1998]:
Grade School Arithmetic (Unlimited Integer/Fract. Precision)
Integers, Rationals, Reals, Complex Numbers (no symbols)
Basic Operations on Numbers: +, -, *, /, ^, \sqrt{}
Factoring Integers: \texttt{FactorInteger, PrimeQ}
Real Numbers (Limited Precision): \texttt{N[num [, ndigits]]}
Complex Numbers: \((a + b \mathbb{I})\)

High School Algebra and Trigonometry
\textit{Symbols Introduced} (A–Z, a–z, α–ω)
Algebraic Manipulation: \texttt{Collect, Expand, Factor, Simplify}
Solving Equations and Systems: \texttt{Solve}
Trigonometry: \texttt{Cos, Cot, Csc, Sec, Sin, Tan}
Hyperbolic Trigonometry: \texttt{Cosh, Coth, Csch, Sech, Sinh, Tanh}
Graphing: \texttt{ContourPlot, ListPlot, Plot, Plot3D, Show}

College Calculus, Differential Equations, and Linear Algebra
\textit{Symbolic Solutions of Derivatives and Integrals}
Integration and Differentiation (Multivariable): \texttt{D, Integrate}
Differential Equations (Linear and Certain Nonlinear): \texttt{DSolve}
Lists, Series and Limits: \{\}, \texttt{Flatten, Limit, Series, Table}
Vectors and Matrices: \texttt{Det, Dot, Eigenvalues, Inverse}

Graduate School
A long list of more complex built-in functions is available along
with interval arithmetic and arithmetic with user-chosen precision.

\textit{Take full advantage of tools like this - use the solution time that will be saved to THINK about the MODEL and what the method’s RESULTS mean! Otherwise, you are contributing very little to the understanding and solution of the problem.}
Interpreted vs. Compiled Programming Languages

**Interpreted Languages (Mathematica)**
Instruction-by-Instruction Processing:
- Syntax Checking
- Translation
- Internal Representation
- “Simulated Execution”
- Read-Evaluate-Print Loop
- Supports Interactive Algorithm Development
Final Program = Interpreter + Application Program
Slower Simulated Execution
Better Diagnostics
Mathematica is a Registered Product of Wolfram Inc.
- Special-Purpose Language with Extra Features
- Expression-Oriented
- For Moderate Sized Numeric, Symbolic, Graphic Problems

**Compiled Languages (C, C++, FORTRAN, Pascal)**
Total Program Processing (No Execution):
- Lexical Analysis
- Syntax Analysis
- Intermediate Code Generation
- Code Optimization (Often Optional)
- Final Code Generation
- Supports Production Code Development
Subsequent Linking and Loading
Faster Execution Done Multiple Times
Many Vendors for Compiled Languages
- General-Purpose Languages
- Statement Oriented
  Good for Large Numeric and Text Processing Problems
Using the Mathematica® Interpreter

- A user normally enters *functions as commands* in *cells* (denoted by blue brackets “]” on the right) in a *notebook* window and executes these commands by using the `<Shift><Enter>` key combination. The results are computed and are normally displayed, unless a semi-colon “;” ends the command.

- The style of the cell determines whether it contains unevaluated text (by not using `<Shift><Enter>`) or commands. Unevaluated text may be formatted (using `Format`) almost as one would do with a word processor.

- The on-line **Help** menu contains all the documentation.

- A command can be split over any number of lines before using `<Shift><Enter>` to evaluate it.

- Use **Kernel → Abort Evaluation** to terminate a lengthy calculation.

- Parentheses “()” are used for grouping, but braces “{ }” only denote lists. Vectors are represented as lists and matrices are nested lists. Use `MatrixForm` for display only, not for a vector or matrix definition.

- A pre-defined command is in the form of a function and has a name, which always begins with an upper-case letter, and has arguments within *brackets* separated by commas. The arguments are passed by “name” (a text-string substitution), not passed by “value” or by “reference.”

- Mathematica is case-sensitive, has many predefined symbols, all start with an upper-case letter or have a Greek symbol (e.g., *Pi* or *π*). Mathematica also has several mathematically-oriented input palettes.

- Variables are common in Mathematica and typically begin with a lower-case letter, but an upper-case letter may also be used. However, the symbols *C*, *D*, *E*, *I*, *N*, *Infinity* and all primitive (built-in) function names are protected. (*C* – speed of light, *D* – differentiate, *E* – base of natural logarithms, *I* – imaginary component, *N* – numeric function).
• The symbol \( / \) (called “slash-dot”) is the preferred method for temporarily assigning values to variables. These are used with rules, defined by using the \( \rightarrow \) (dash-angle), which turns into the \( \rightarrow \) symbol. Using the \( = \) symbol instead of \( / \). for temporary assignments will often create difficulties. For example, \( \text{Cos}[t] / \). \( t \rightarrow 2\pi \) yields 1.

• User-defined functions with arguments should typically be defined with a set-delayed assignment symbol (\( : = \)); use the (\( = \)) with no arguments.

• User-defined functions may be defined with and without arguments. Using the form \( f = \text{fundef} \) is best for manipulation, finding analytic derivatives, simplification, and the like. This form may also be plotted. This form may be evaluated by using temporary variable assignments through the use of \( / \). Using the form \( f[u_, v_, w_, \ldots] := \text{fundef} \) is best for evaluation (e.g., \( f[10, 20, 30, \ldots] \)), expression substitution, and the like. This form may also be plotted.

• The \texttt{Plot} function should be used to graphically display all functions of one variable. It is suggested that the option \texttt{PlotRange \rightarrow All}, always be used to insure that all of the function is plotted in the specified domain and not just the “interesting” part. It is suggested that the options be understood and appropriately used (use \texttt{Options[Plot]} to see them). Each plot may be saved by assigning the returned graphics information to a variable, and then combined later by using \texttt{Show}.

• The \texttt{Plot3D} function should be used to graphically display all functions of two variables. It is suggested that the option \texttt{PlotRange \rightarrow All}, always be used to insure that all of the function is plotted in the specified domain and not just the “interesting” part. It is suggested that the options be understood and appropriately used. Each plot may be saved by assigning the returned graphics information to a variable, and then combined later by using \texttt{Show}.

• The \texttt{ContourPlot} function should also be used to graphically display functions of two variables. Sometimes more insight is gained here than with \texttt{Plot3D}. It is suggested that the options be understood and appropriately used.
• The **Module** construct should be used when writing your own function programs in order to **protect** all new symbols (e.g., variables) and make them local.

• To print the cell brackets, so that inputs and outputs can be clearly identified, use **File ➔ Printing Settings ➔ Printing Options...** and place a checkmark next to **Print cell brackets**, then click the **OK** button.
Some Useful Mathematica® Kernel Functions
(Also See Function Palettes. Use ? or ?? or the Help Browser.)

Abs[]    Append[]
Apply[]   ArcCos[]
Array[]   Ceiling[]
Clear[]   Collect[]
ConstrainedMax[]
ContourPlot[]
D[]       Det[]
Do[]      Drop[]
DSolve[]
Eigenvectors[]
Expand[]
FindMinimum[]
First[]    Flatten[]
Floor[]    FullSimplify[]
IdentityMatrix[]
Insert[]   Integrate[]
Inverse[]
LinearProgramming[]
ListPlot[]
Map[]      MatrixForm[]
Max[]      Min[]
Mod[]      Module[]
N[]        NDSolve[]
Needs[]    Normal[]
NSolve[]   Outer[]
ParametricPlot[]
Plot[]     Plot3D[]
Prepend[]
Round[]    Series[]
Show[]     Simplify[]
Sin[]      Solve[]
Sort[]     Sqrt[]
StringLength[]
Table[]    TableForm[]
Tan[]      Timing[]
Transpose[]

Special Symbols  +, -, *, /, ^, =, ==, !=, <, >, <=, >=, (), {}, [], &&, ||, ;,
Numerical Computation with Mathematica

Numerical computation can be done in the following ways:

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>Arbitrary precision whole-number computations; invoked when whole numbers are used. <em>Best when exact values are needed.</em></td>
</tr>
<tr>
<td>Rational</td>
<td>Arbitrary precision fraction computations; invoked when a number of the form Integer/Integer is used. <em>Best for exact values.</em></td>
</tr>
<tr>
<td>Float</td>
<td>Limited precision (faster) computations with a limited range; invoked when a decimal point is used in a number. <em>Best for most computations.</em> $\text{MachinePrecision}$ gives this precision (e.g., 16). $\text{MinMachineNumber}$ is the smallest positive float (e.g., $2.22507 \times 10^{-308}$). $\text{MaxMachineNumber}$ is the largest positive float (e.g., $1.79769 \times 10^{308}$).</td>
</tr>
<tr>
<td>Arbitrary</td>
<td>Arbitrary precision (slower) floating-point computations with a limited range; invoked when $\text{N[Expr, p]}$ or $\text{N[RatExpr, p]}$ is used. <em>Best for investigating the numerical properties of a new kind of calculation.</em> $\text{MinNumber}$ is the smallest positive arbitrary precision number (e.g., $1.05934595 \times 10^{-323228015}$). $\text{MaxNumber}$ is the largest positive arbitrary precision number (e.g., $1.44039712 \times 10^{323228010}$).</td>
</tr>
</tbody>
</table>

Note: All of these numbers can also be used in *complex arithmetic* and *interval arithmetic*.