Introduction of Computer-Aided Nano Engineering

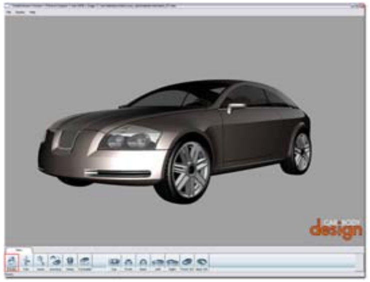
Prof. Yan Wang
Woodruff School of Mechanical Engineering
Georgia Institute of Technology
Atlanta, GA 30332, U.S.A.
yan.wang@me.gatech.edu

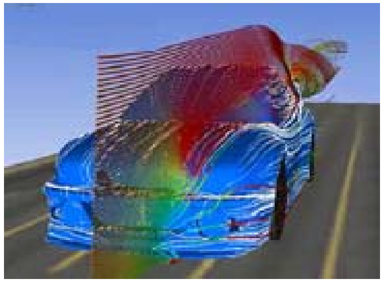
Topics

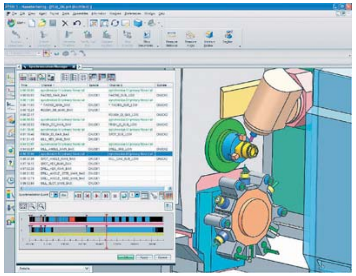
- Computational Nano Engineering
- Modeling & Simulation (M&S)
- Approximations in M&S

Computational Nano-Engineering

- Extensive applications of CAD/CAM/CAE software tools in traditional manufacturing lead to
  - Scalable processes
  - Cost effective and high-quality products with short time-to-market


(a) CAD


(b) CAE


(c) CAPP/CAM


(d) CAM

- ***Virtual Prototyping*** at *nanoscales*
  - Computer-Aided Nano-Design (CAND)
  - Computer-Aided Nano-Manufacturing (CANM)
  - Computer-Aided Nano-Engineering (CANE)

Computational Nano-Engineering

- Use Modeling & Simulation tools to systematically resolve the issue of "lack of design" at nano scales.

PAST: Discovery-Based Science and Product Development

Discover novel Nanostructures, nanoparticles, and nanomaterials through investigator-initiated exploratory research on a broad range of materials

Determine nanomaterial properties (chemical, physical, and biological)

Identify potential applications of value

Assess commercial viability

Nanomaterials enter limited markets

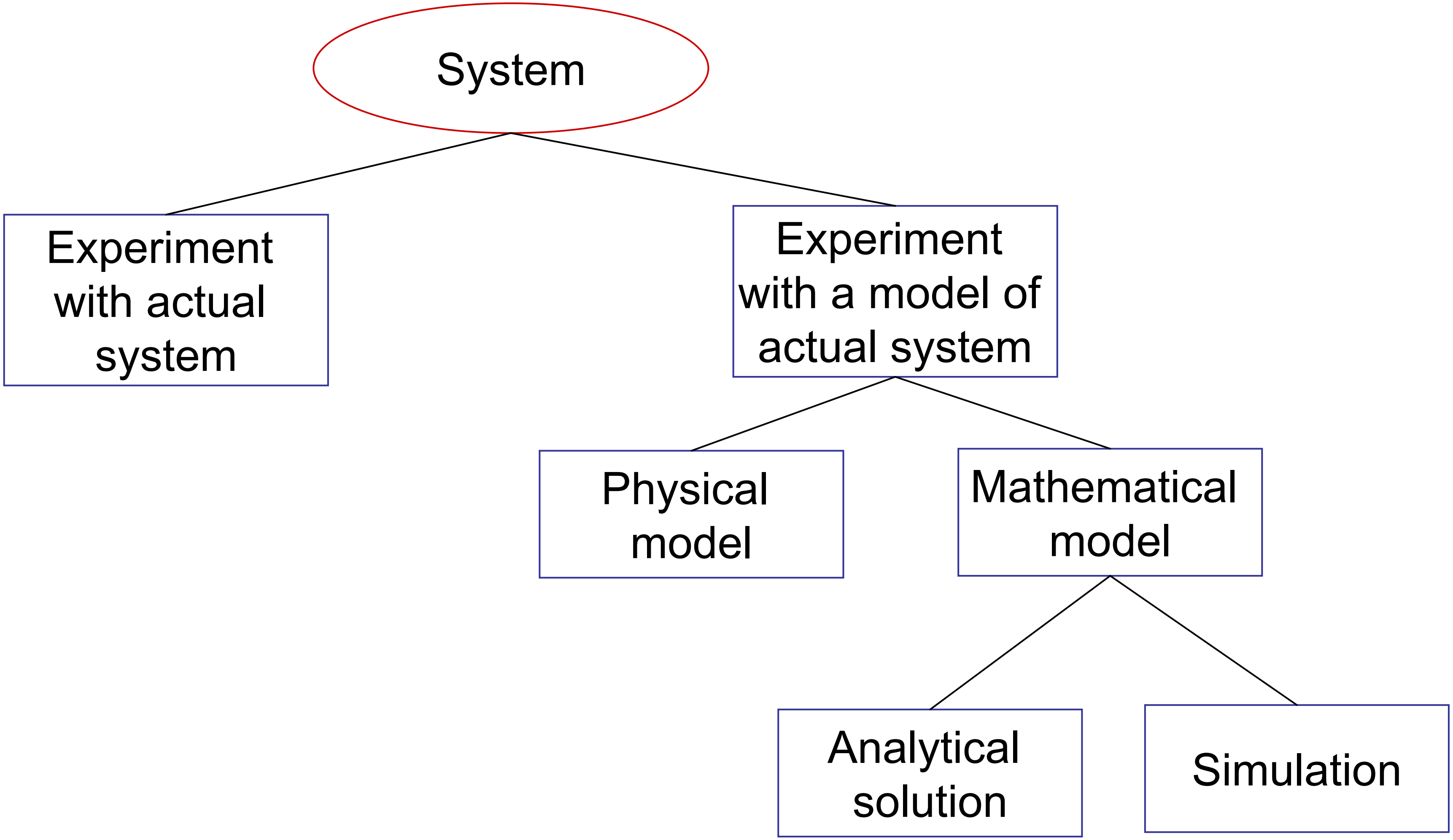
FUTURE: Application-Based Problem Solving

Start with existing needs, problems, or challenges in end-use application

Design, produce, and scale up nano-based materials with exact properties needed (based on established understanding and methods)

Large numbers of diverse products based on Nanomaterials By Design rapidly enter multiple markets

How to study a system



What is Modeling?

Why mathematical modeling?

- Advantages

- Disadvantages

An example of modeling

- Free fall model

Mathematical model

- *Dependent variable=f(Independent variable)*

$$y = f(x)$$

- High dimensional

$$y = f(x_1, x_2, \ldots)$$

- Parametric systems

$$y = f\left(x_1(u), x_2(u), \ldots, x_n(u), u\right)$$

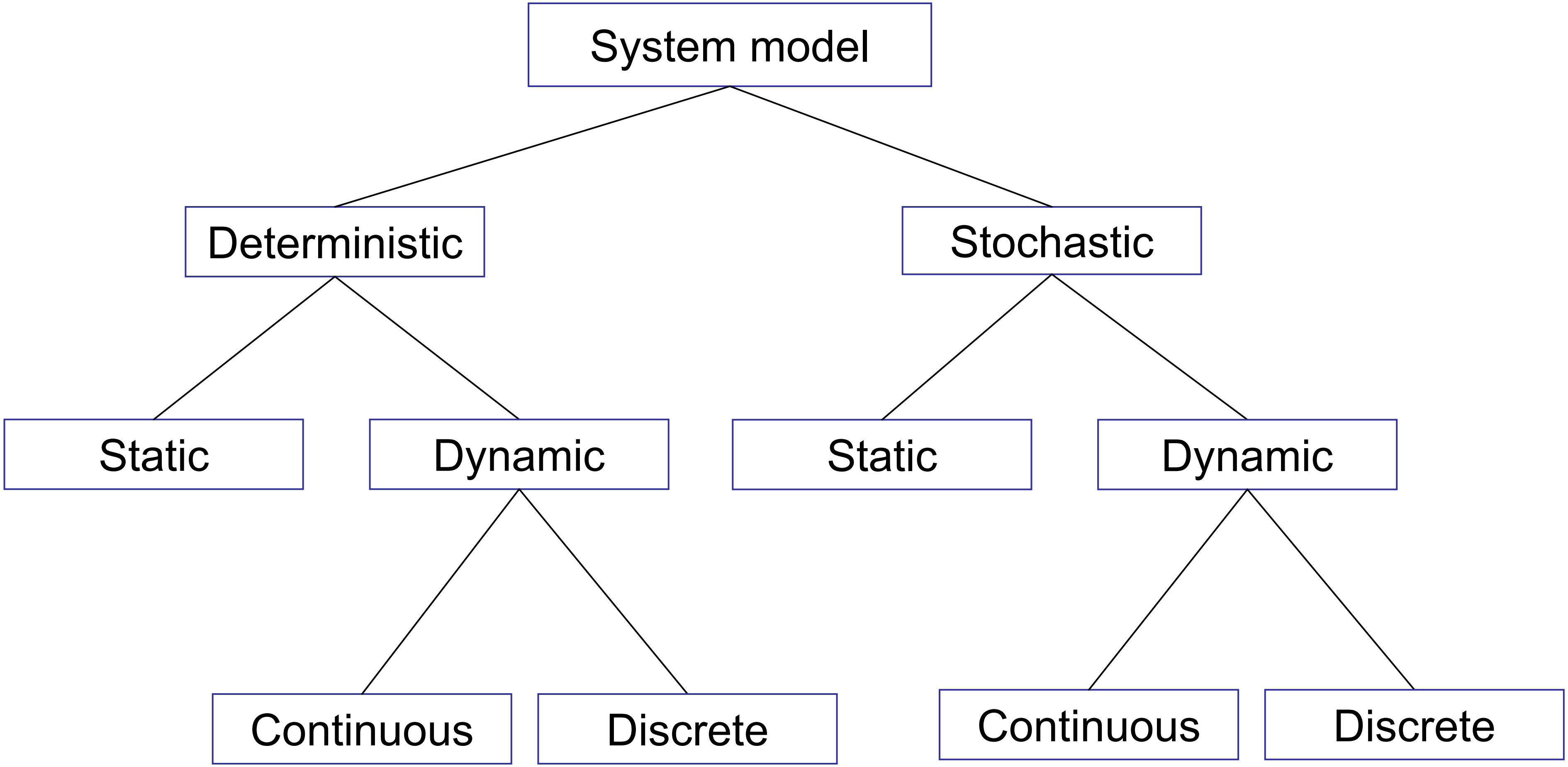
- "Noisy" systems

$$y = f\left(x, \gamma\right)$$

Complexity of Mathematical Models

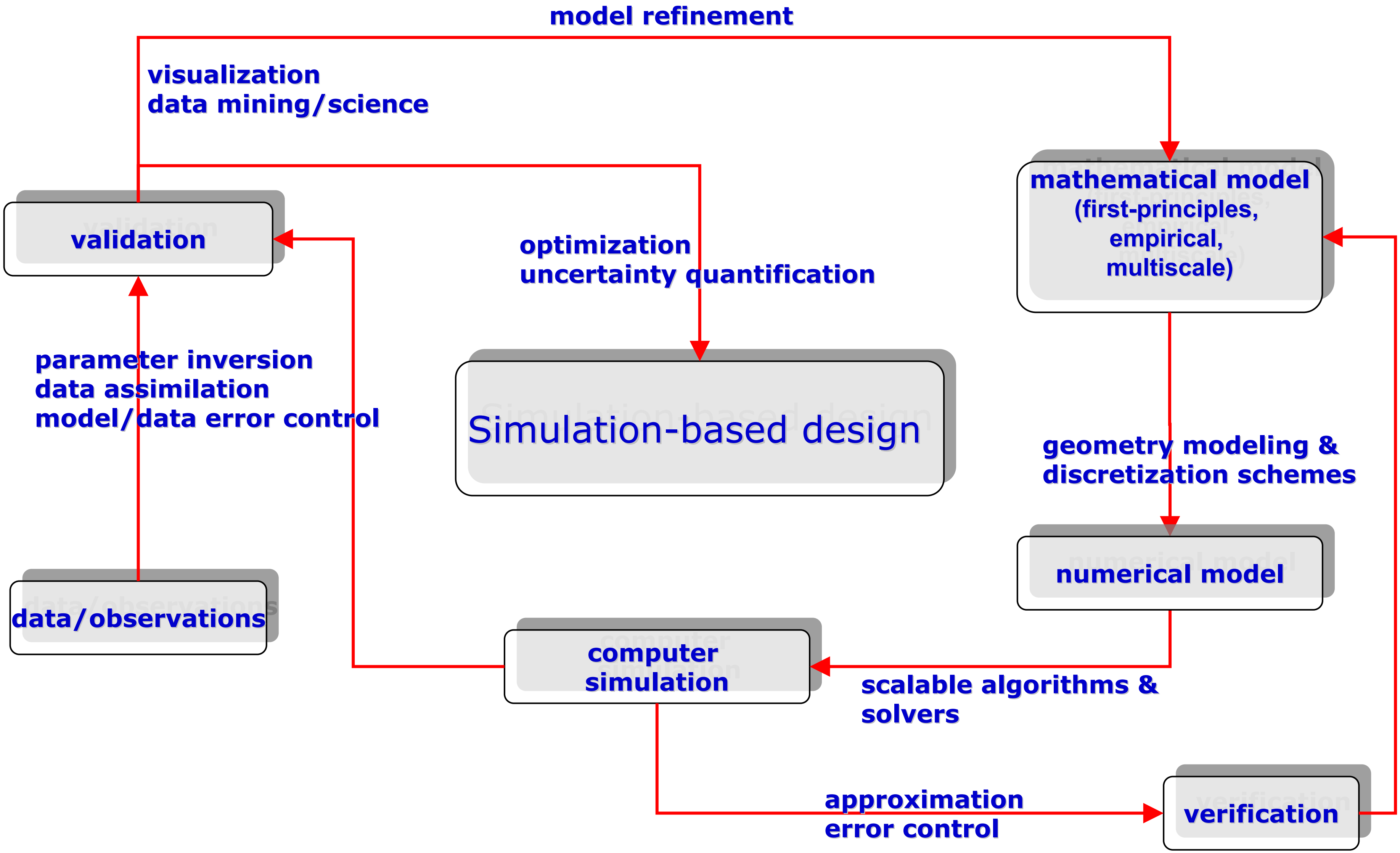
Simple	Linear	Algebraic Equation / Closed-form	Static
↓	↓	↓	↓
Complex	Nonlinear	Differential Equation	Dynamic

Model Taxonomy

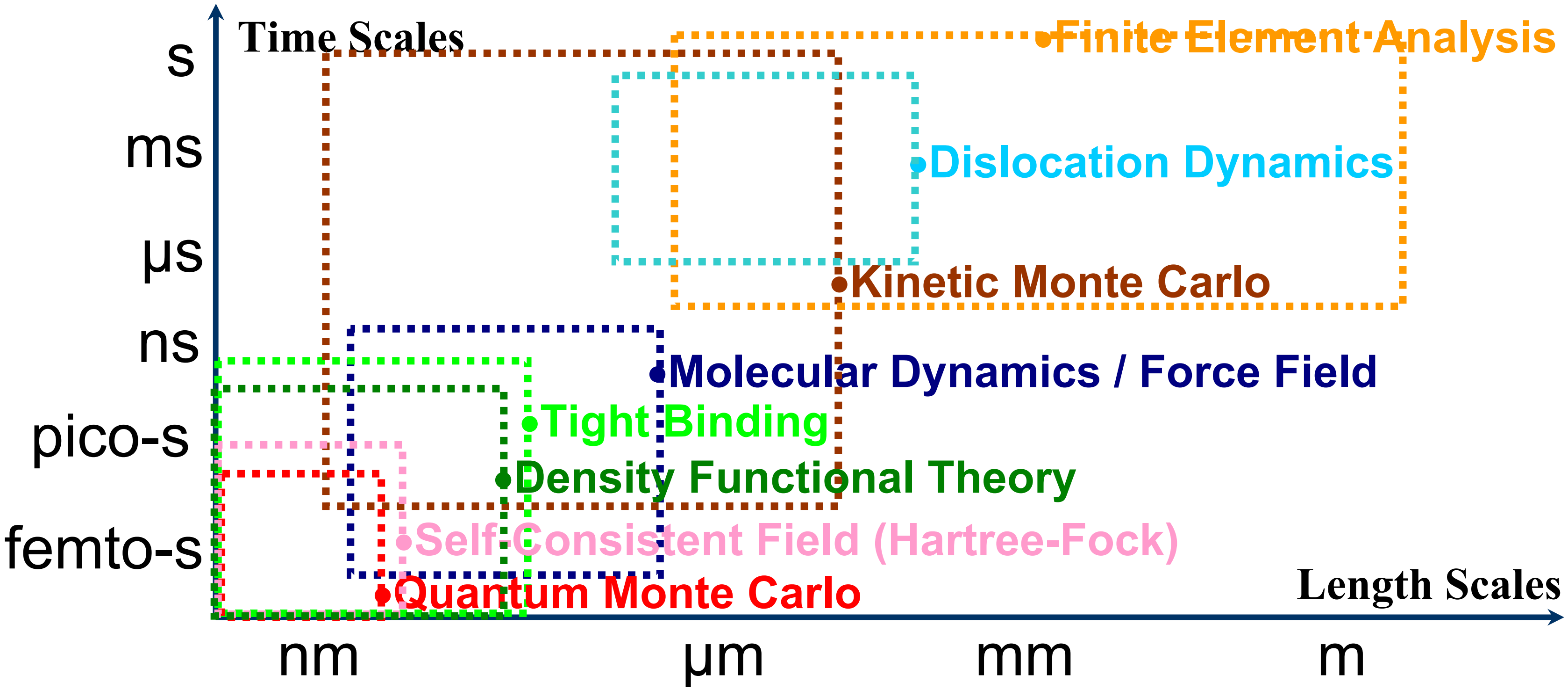


Georgia Tech

Simulation-based Design



Modeling & Simulation at Multiple Scales



Various methods used to simulate at different length and time scales

Approximations in simulation

- **mathematical models ➔ numerical models**
  - **Taylor series**
  - **Functional analysis**
- numerical models ➔ computer codes
  - Discretization (differentiation, integration)
  - Searching algorithms (solving equations, optimization)
  - Floating-point representation

Mathematical models → Numerical models
Approximation in Taylor Series

- Truncation

Mathematical models → Numerical models
Functional Analysis

- Convert complex functions into simple and computable ones by transformation in vector spaces
  - Fourier analysis
  - Wavelet transform
  - Polynomial chaos expansion
  - Spectral methods
  - Mesh-free methods
  - …

Mathematical models → Numerical models Functional Analysis

- Approximate the original $f(x)$ by linear combinations of basis functions $\psi_i(x)$'s as

$$f(x) \approx \sum_{i=0}^{N} c_i \psi_i(x)$$

- In a vector space (e.g. Hilbert space) with an infinite number of dimensions

$$f(x) = \sum_{i=0}^{\infty} c_i \psi_i(x)$$

Mathematical models → Numerical models
Functional Analysis

- An *inner product* $\langle f, g \rangle$ is defined as a "projection" in the vector space, such as

$$\langle f, g \rangle := \int_{-\infty}^{\infty} f(x) g(x) W(x) dx$$

- Typically we choose orthogonal basis functions $\psi_i(x)$'s such that

$$\langle \psi_i, \psi_j \rangle = \int_{-\infty}^{\infty} \psi_i(x) \psi_j(x) W(x) dx = \begin{cases} Constant & (\forall i, j, i = j) \\ 0 & (i \neq j) \end{cases}$$

for orthonormal basis functions

$$\langle \psi_i, \psi_j \rangle = \int_{-\infty}^{\infty} \psi_i(x) \psi_j(x) W(x) dx = \begin{cases} 1 & (i = j) \\ 0 & (i \neq j) \end{cases}$$

Mathematical models → Numerical models
Functional Analysis

- The coefficients c_i's are computed by

$$c_i = \frac{\langle f, \psi_i \rangle}{\langle \psi_i, \psi_i \rangle}$$

- The computable function is

$$f(x) \approx \sum_{i=0}^{N} c_i \psi_i(x)$$

with ***truncation***!

Functional Analysis – Fourier Series Approximations

$$f(t) = a_0 + a_1\cos(\omega_0 t) + b_1\sin(\omega_0 t) + a_2\cos(2\omega_0 t) + b_2\sin(2\omega_0 t) + \cdots$$
$$= a_0 + \sum_{k=1}^{\infty}\left[a_k\cos(k\omega_0 t) + b_k\sin(k\omega_0 t)\right]$$

where $\omega_0 = \frac{2\pi}{T}$ is called the *fundamental frequency*.

- All *periodic functions* can be approximated by Fourier Series well!
- Is used in ***plane-wave density functional theory*** simulations
- Because of efficient Fast Fourier Transform (FFT)!

Functional Analysis – Fourier Series Approximations

- Inner products

$$a_0 = \frac{1}{T}\int_0^T f(t)\,dt$$

$$a_k = \frac{2}{T}\int_0^T \cos(k\omega_0 t) f(t)\,dt \qquad (k = 1, 2, \ldots)$$

$$b_k = \frac{2}{T}\int_0^T \sin(k\omega_0 t) f(t)\,dt$$

Fourier Series	*Fourier Transform*	*Discrete Fourier Transform*
$\tilde{c}_k = \frac{1}{T}\int_0^T f(t)\,e^{-ik\omega_0 t}dt$	$\tilde{F}(i\omega_0) = \int_{-\infty}^{\infty} f(t)\,e^{-i\omega_0 t}dt$	$\tilde{F}_k = \sum_{n=0}^{N-1} f_n e^{-ik\omega_0 n} \quad (k = 0, \ldots, N-1)$
$f(t) = \sum_{k=-\infty}^{\infty} \tilde{c}_k e^{ik\omega_0 t}$	$f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \tilde{F}(i\omega_0)\,e^{i\omega_0 t}d\omega_0$	$f_n = \frac{1}{N}\sum_{k=0}^{N-1} \tilde{F}_k e^{ik\omega_0 n} \quad (n = 0, \ldots, N-1)$

- Computational complexity: $O(N^2)$
- FFT Complexity: $O(N\log_2 N)$

Functional Analysis – Wavelet Transform

- *Wavelet basis functions*

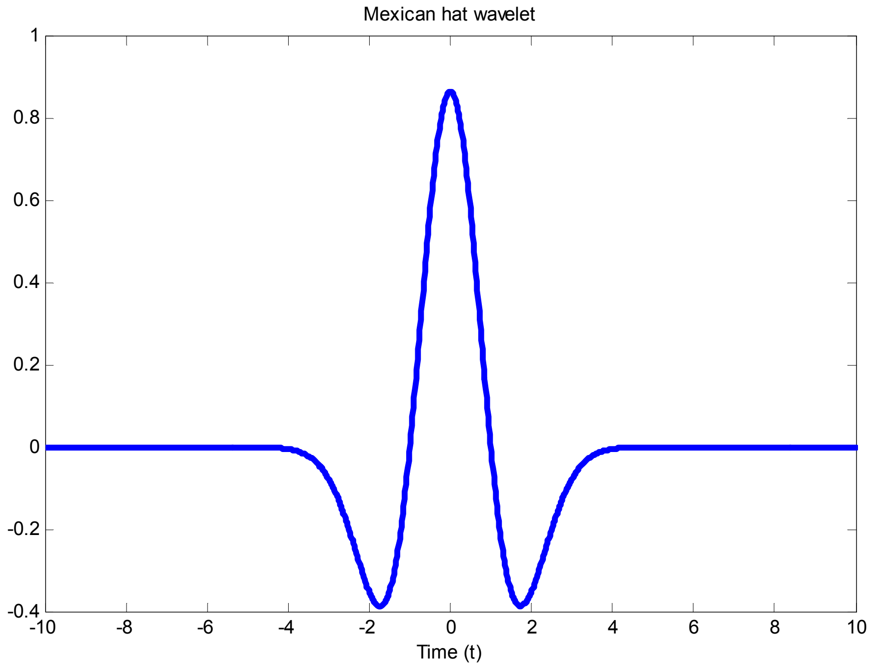
$$\psi_{a,b}(x) = \frac{1}{\sqrt{a}}\psi\left(\frac{x-b}{a}\right)$$

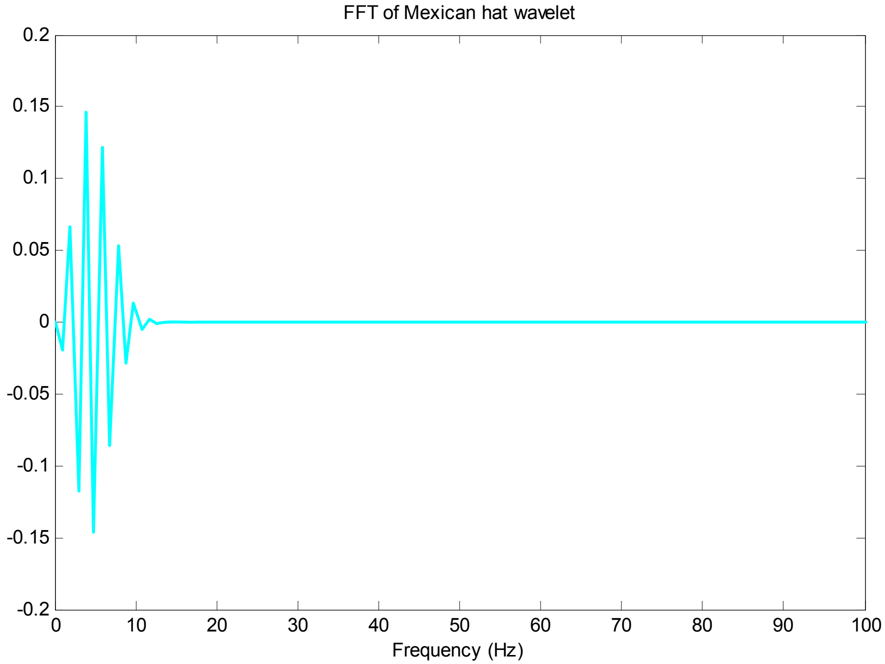
where $\psi(x)$ is a continuous function in *both* the real and reciprocal spaces called *mother wavelet*, a is the scale factor, and b is the translation factor.

- $\psi(x)$ satisfies
  - Admissibility
  - Regularity condition

Example Continuous Wavelet Functions - Mexican hat

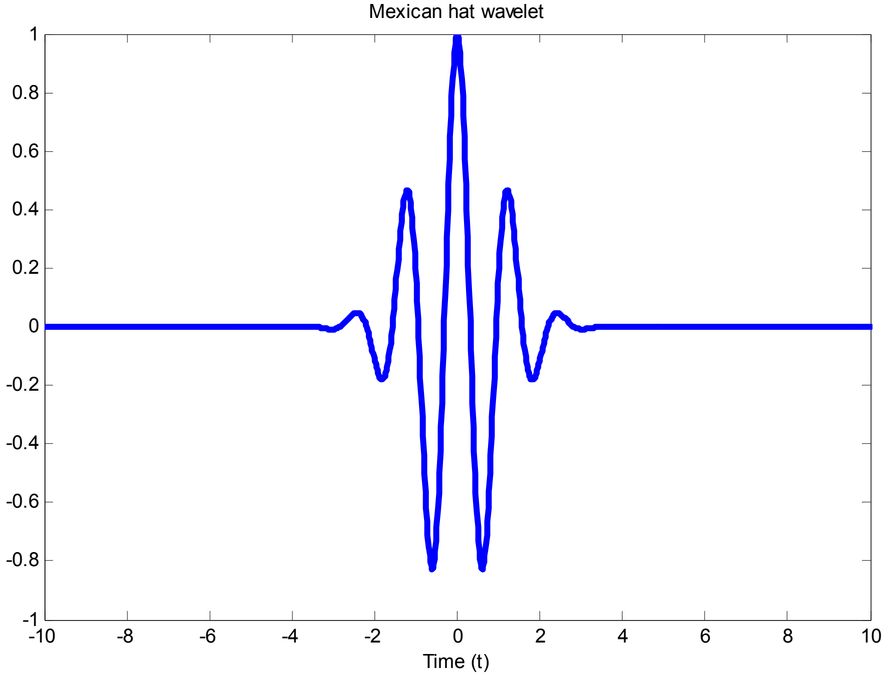
$$\psi(x) = \frac{2}{\sqrt{3}} \pi^{-1/4} \left(1 - x^2\right) e^{-x^2/2}$$

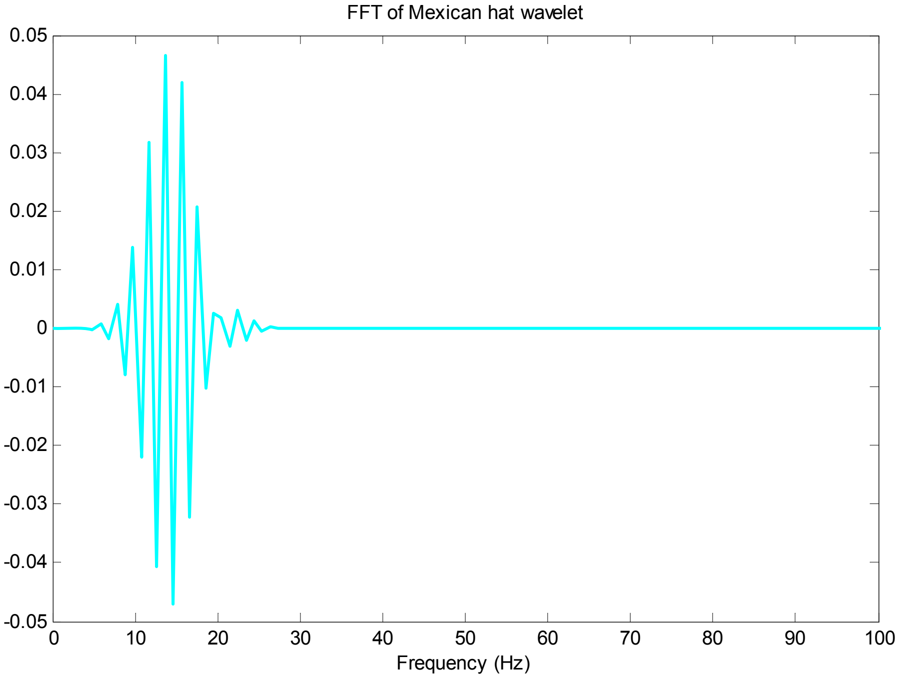




Example Continuous Wavelet Functions - Morlet

$$\psi(x) = e^{-x^2/2}\cos\left(5x\right)$$





Functional Analysis – Wavelet Transform

- Continuous Transform

$$\tilde{f}\left(a,b\right)=\int_{-\infty}^{+\infty} f(x)\psi_{a,b}^{*}(x)dx$$

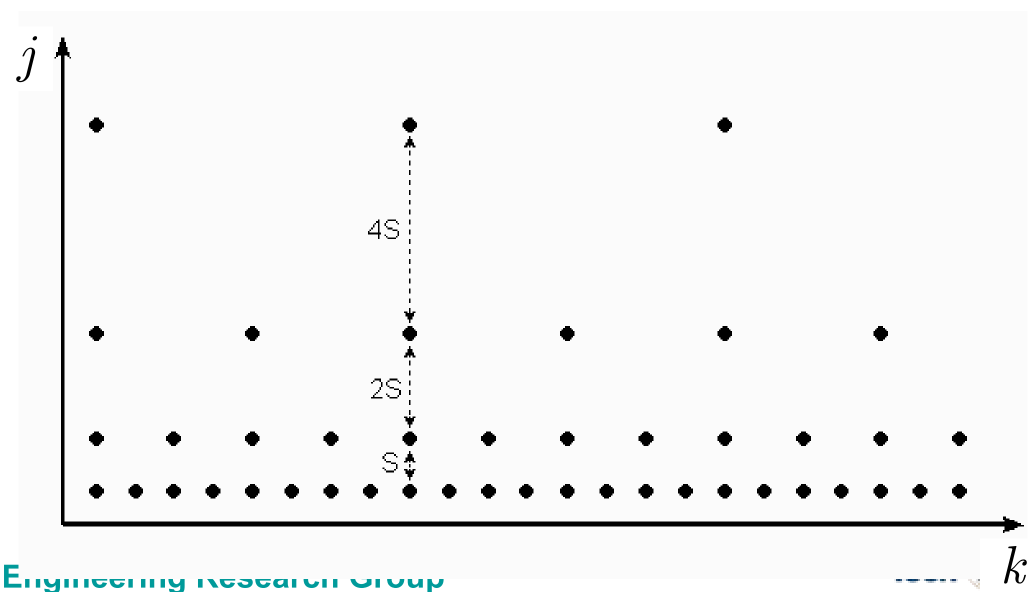
where $\psi_{a,b}^{*}(x)$ is the complex conjugate of $\psi_{a,b}(x)$.

- Inverse Transform

$$f\left(x\right)=\frac{1}{c_{\psi}}\int_{-\infty}^{+\infty}\int_{0}^{+\infty}\tilde{f}(a,b)\psi_{a,b}(x)\frac{da}{a^{2}}db$$

- Discrete Transform

$$\psi_{2^j,k}\left(x\right)=\frac{1}{\sqrt{2^j}}\psi\left(\frac{x-k}{2^j}\right)$$



Functional Analysis – Wavelet Transform

- **Admissibility**

$$\int_0^{+\infty} \frac{\left|\hat{\psi}(\omega)\right|^2}{\left|\omega\right|} d\omega < \infty$$

where $\hat{\psi}\left(\omega\right) = \int_{-\infty}^{\infty} \psi\left(x\right) e^{-i\omega x} dx$ is the Fourier transform of $\psi(x)$.

- This implies $\left|\hat{\psi}(\omega = 0)\right|^2 = 0$ or equivalently $\int_{-\infty}^{\infty} \psi\left(x\right) dx = 0$
  "No information loss in reconstruction"
- "must be a wave with zero mean" -- **wave-**

Functional Analysis – Wavelet Transform

Regularity condition

$$\tilde{f}\left(a, b=0\right) \approx \frac{1}{\sqrt{a}}\left[\sum_{k=0}^{N}\frac{1}{k!}f^{(k)}(0)\int x^k\psi\left(\frac{x-0}{a}\right)dx + O(x^{k+1})\right]$$
$$= \frac{1}{\sqrt{a}}\left[\sum_{k=0}^{N}\frac{1}{k!}f^{(k)}(0)M_k a^{k+1} + O(a^{k+2})\right]$$
$$= \frac{1}{\sqrt{a}}\left[\frac{1}{0!}f(0)M_0 a + \frac{1}{1!}f^{(1)}(0)M_1 a^2 + \ldots + \frac{1}{N!}f^{(N)}(0)M_N a^{N+1} + O(a^{k+2})\right]$$

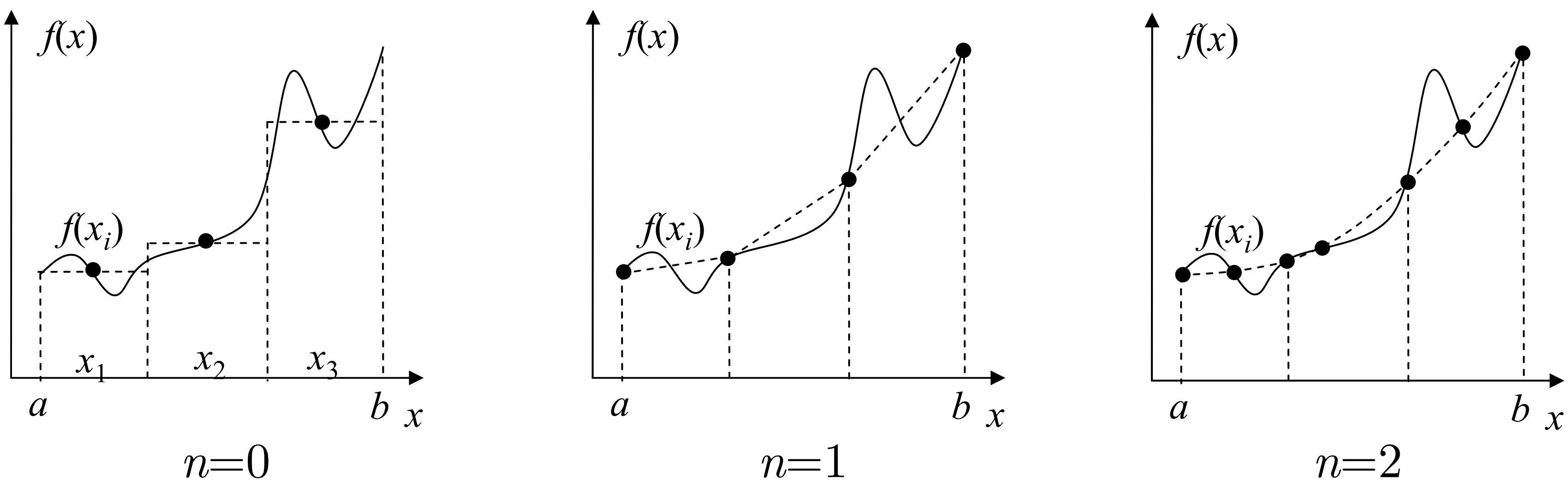
- wavelet functions should have some smoothness and concentration in both time and frequency domains
- vanishing moments M_1, …, M_N as the scale factor a increases (admissibility: M_0=0)
- "Fast Decay" -- **-let**

Approximations in M&S

- mathematical models ➔ numerical models
  - Taylor series
  - Functional analysis
- **numerical models ➔ computer codes**
  - **Discretization (differentiation, integration)**
  - **Searching algorithms (solving equations, optimization)**
  - **Floating-point representation**

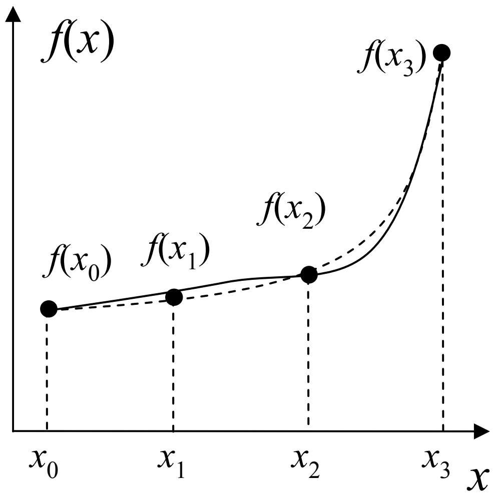
Numerical Model → Computer Code
Compute integrals

- Quadrature
  - Approximate the integrand function by a polynomial of certain degree



Numerical Model → Computer Code
Compute integrals

- Quadrature
  - Approximate the integral by the weighted sum of regularly sampled functional values
    - e.g. Simpson's 3/8 rule



$$I \approx \int_{x_0}^{x_3} f^{(3)}(x)\,dx = \frac{3h}{8}\left[f(x_0) + 3f(x_1) + 3f(x_2) + f(x_3)\right]$$

Numerical Model → Computer Code
Compute integrals

- Monte Carlo simulation
  - Let $p(u)$ denote uniform density function over $[a,\ b]$
  - Let U_i denote i th uniform random variable generated by Monte Carlo according to the density $p(u)$
  - Then, for "large" N

$$\int_a^b f(x)dx \approx \frac{b-a}{N}\sum_{i=1}^{N} f(U_i)$$

  - Variance reduction (importance sampling) to improve efficiency

Numerical Model → Computer Code
Compute derivatives

- Finite-divided-difference methods
  - Approximated derivatives come from Taylor series
    - e.g. forward-finite-difference

$$f(x_{i+1}) = f(x_i) + f'(x_i)(x_{i+1} - x_i) + O(h^2) = f(x_i) + f'(x_i)h + O(h^2)$$

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_i)}{h} + O(h)$$

Floating-Point Representation

- How does computer represent numbers?

Perfect world

Imperfect world

Ariane 5

- Exploded 37 seconds after liftoff
- Cargo worth $500 million

- Why
  - Computed horizontal velocity as floating point number
  - Converted to 16-bit integer
  - Worked OK for Ariane 4
  - Overflowed for Ariane 5
    - Used same software

Do you trust your computer?

Rump's function:

$$f(x, y) = 333.75y^6 + x^2(11x^2y^2 - y^6 - 121y^4 - 2) + 5.5y^8 + \frac{x}{2y}$$

$$f(x = 77617, y = 33096) = ?$$

- Single precision: f = 1.172603…
- Double precision: f = 1.1726039400531…
- Extended precision: f = 1.172603940053178…
- Correct one is: f = -0.827396059946821**3**

Bug?

Defect?

Another Story

- On February 25, 1991
- A Patriot missile battery assigned to protect a military installation at Dhahran, Saudi Arabia
- But … failed to intercept a Scud missile
- 28 soldiers died
- … an error in computer arithmetic

$$0.1 \times 10 \neq 1$$

IEEE 754 Standard

IEEE Floating Point Representation

s	exponent E	mantissa T
1 bit	w bits 8 bits	23 bits $t = p - 1$ bits

IEEE Double Precision Floating Point Representation

1 bit	11 bits	52 bits
s	exponent	mantissa

- If $E = 2^w - 1$ and $T \neq 0$, then v is **NaN** regardless of S.
- If $E = 2^w - 1$ and $T = 0$, then $v = (-1)^S \times \infty$.
- If $1 \leq E \leq 2^w - 2$, then $v = (-1)^S \times 2^{E - bias} \times (1 + 2^{1-p} \times T)$;

normalized numbers have an implicit leading significand bit of 1.

- If $E = 0$ and $T \neq 0$, $v = (-1)^S \times 2^{emin} \times (0 + 2^{1-p} \times T)$;

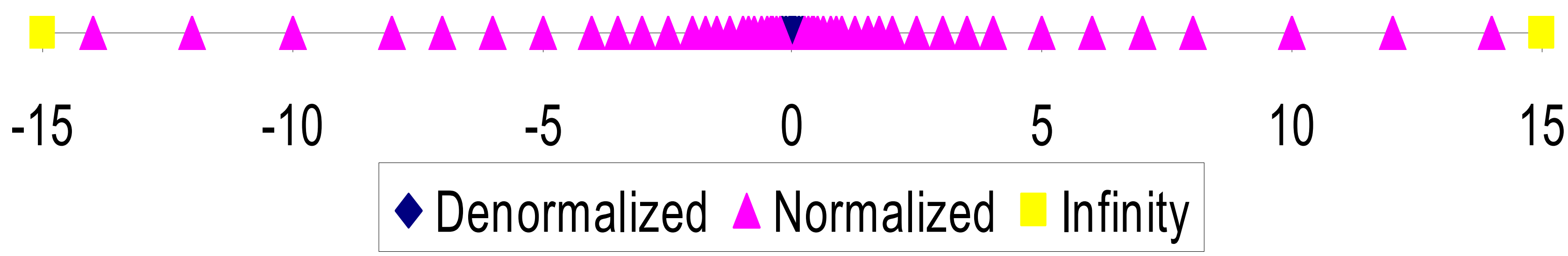
denormalized numbers have an implicit leading significand bit of 0.

- If $E = 0$ and $T = 0$, then $v = (-1)^S \times 0$ (signed zero)

where $bias = 2^{w-1} - 1$ and $emin = 2 - 2^{w-1} = 1 - bias$

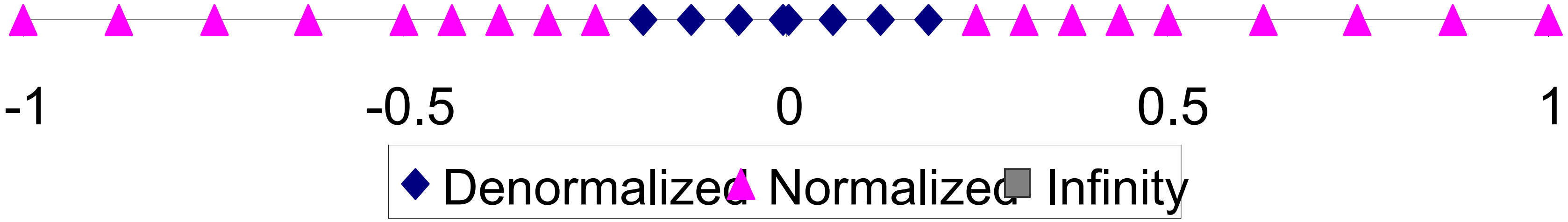
Distribution of Values

- 6-bit IEEE-like format
  - w = 3 exponent bits
  - t = 2 fraction/mantissa bits
  - bias = 3



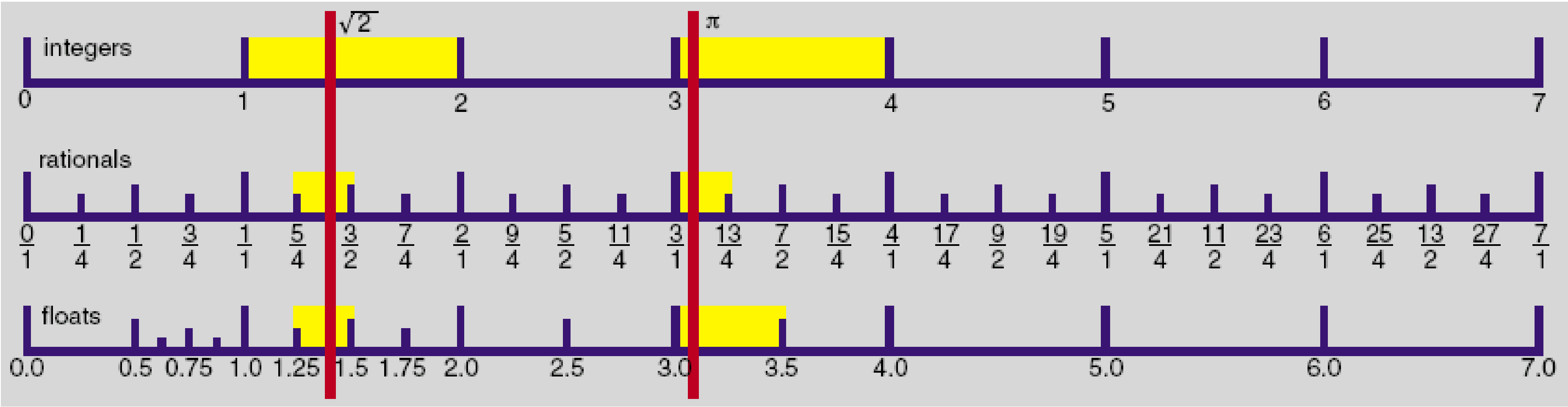
Distribution of Values (zoom-in view)

- 6-bit IEEE-like format
  - w = 3 exponent bits
  - t = 2 fraction/mantissa bits
  - bias = 3



Georgia Tech

Round-Off Errors



- Overflow error – "not large enough"
- Underflow error – "not small enough"
- Rounding error – "chopping"

- http://www.cs.utah.edu/~zachary/isp/applets/FP/FP.html

Special Numbers

Expression	Result
0.0 / 0.0	NaN
1.0 / 0.0	Infinity
-1.0 / 0.0	-Infinity
NaN + 1.0	NaN
Infinity + 1.0	Infinity
Infinity + Infinity	Infinity
NaN > 1. 0	false
NaN == 1.0	false
NaN < 1.0	false
NaN == NaN	false
0.0 == -0.0	true

- standard range of values permitted by the encoding (from 1.4e-45 to 3.4028235e+38 for float)

Floating point Hazards

This expression	does NOT equal to this expression	when
0.0 - f	-f	f is 0
f < g	! (f >= g)	f or g is NaN
f == f	true	f is NaN
f + g – g	f	g is infinity or NaN

```
double s=0;
for (int i=0; i<26; i++) s += 0.1;
System.out.println(s);
```

```
double d = 29.0 * 0.01;
System.out.println(d);
System.out.println((int) (d * 100));
```

- The result is
2.6000000000000001

- The result is
0.29
28

Comparing Floating Point Numbers

- Try to avoid floating point comparison directly
- Testing if a floating number is greater than or less than zero is even risky.
- *Instead*, you should compare the absolute value of the difference of two floating numbers with some pre-chosen epsilon value, and test if they are "close enough"
- If the scale of the underlying measurements is unknown, the test "abs(a/b - 1) < epsilon" is more robust.
- Don't use floating point numbers for exact values

Uncertainties in M&S

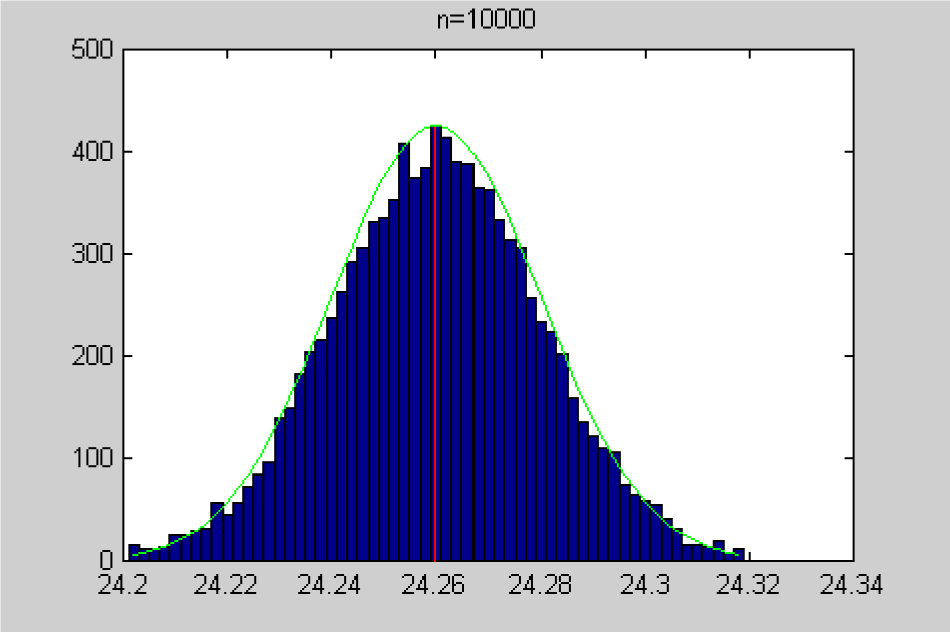
- Model errors due to approximations in truncation or sampling
  - Taylor approximation
  - Functional analysis
- Numerical errors due to floating-point representation
  - Round-off errors

Two types of "uncertainties"

- Aleatory uncertainty (variability, irreducible uncertainty, random error)
  - inherently associated with the randomness/fluctuation (e.g. environmental stochasticity, inhomogeneity of materials, fluctuation of measuring instruments)
  - can only be reduced by taking average of multiple measurements.
- Epistemic uncertainty (incertitude, reducible uncertainty, systematic error)
  - imprecision comes from scientific ignorance, inobservability, lack of knowledge, etc.
  - can be reduced by additional empirical effort (such as calibration).

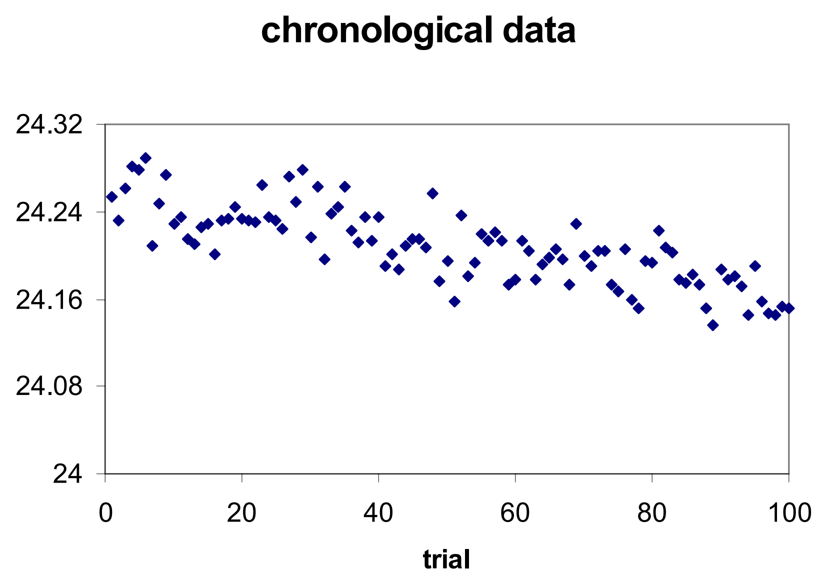
Random Error

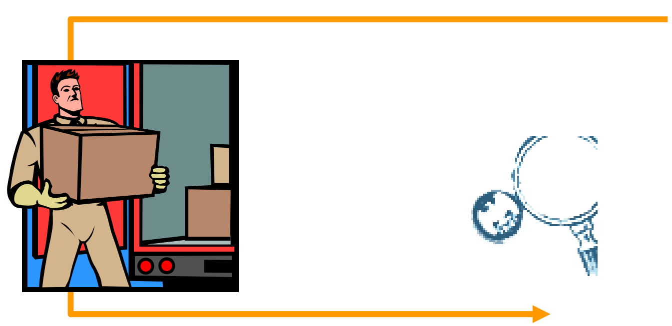
- Determines the *precision* of any measurement
- *Always* present in every physical measurement
  - Better apparatus
  - Better procedure
  - Repeat
- Estimate



Systematic Error

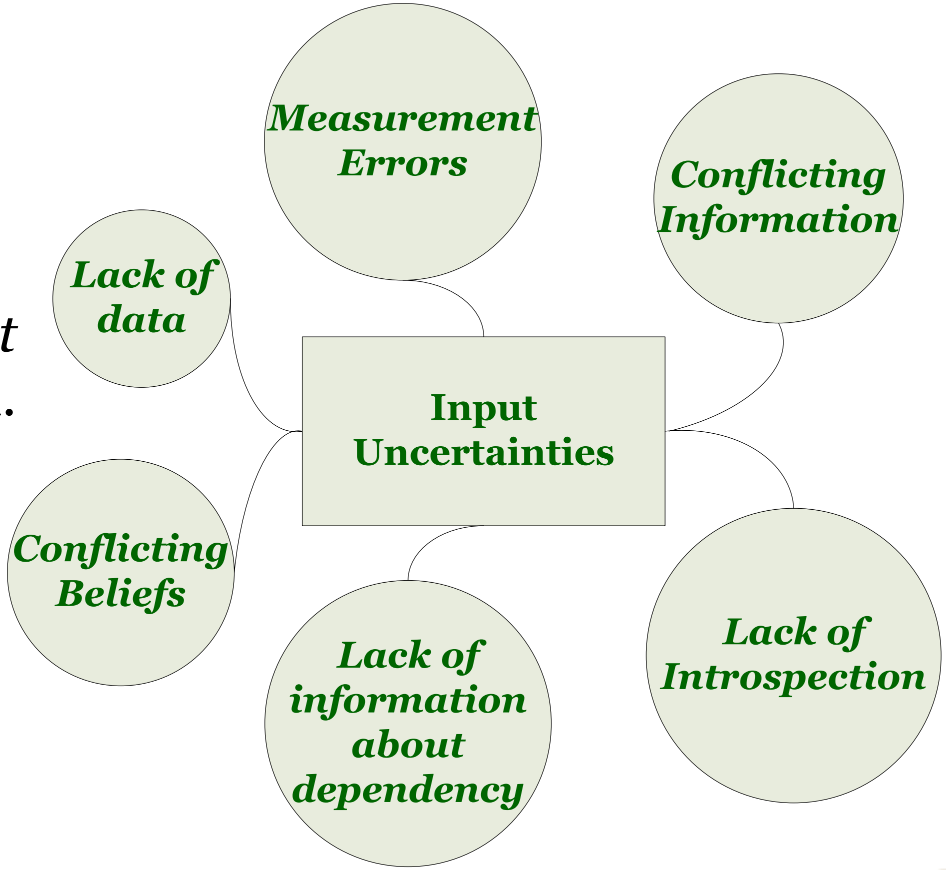
- Determines the *accuracy* of any measurement
- *May* be present in every physical measurement
  - calibration
  - uniform or controlled conditions (e.g., avoid systematic changes in temperature, light intensity, air currents, etc.)
- Identify & eliminate or reduce





Uncertainties in Modeling & Simulation

- ***Aleatory Uncertainty:***
  - *inherent randomness* in the system. Also known as:
    - stochastic uncertainty
    - variability
    - irreducible uncertainty
- ***Epistemic Uncertainty:***
  - due to *lack of perfect knowledge* about the system. Also known as:
    - Incertitude
    - system error
    - reducible uncertainty



Summary

- Modeling is *abstraction*
- M&S ***always*** has *approximations* involved, which are important sources of epistemic uncertainty.
- Computer tricks us

Abstraction
Approximation
Round-off

Further Readings

- Goldberg, D. (1991) "What every computer scientist should know about floating-point arithmetic," *ACM Computing Surveys*, **23**(1), 5-48