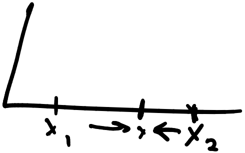
Proof Intuition for Interpolation Error Bound

Let us consider an interpolant $\tilde{f}$ based on $n = 2$ points so

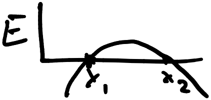
$$\tilde{f}(x_1) = f(x_1) \quad \text{and} \quad \tilde{f}(x_2) = f(x_2).$$

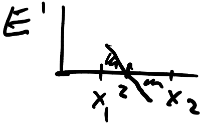
The interpolation error is $O((x_2 - x_1)^2)$ for any $x \in [x_1, x_2]$, why?

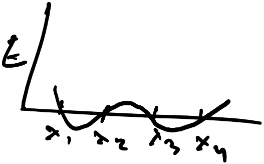


$$E(x) = f(x) - \tilde{f}(x)$$

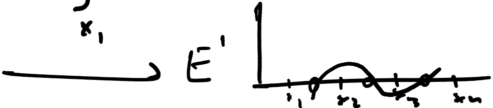
$$E(x_1) = 0 \qquad E(x_2) = 0$$







$$\int_{x_1}^{x_2} E'(x)\,dx = E(x_2) - E(x_1) = 0$$



Proof of Interpolation Error Bound

We can use induction on n to show that if $E(x) = f(x) - \tilde{f}(x)$ has n zeros $x_1, \ldots, x_n$ and $\tilde{f}$ is a degree n polynomial, then there exist $y_1, \ldots, y_n$ such that

$\in [x_1, x_n]$

$$E(x) = \int_{x_1}^{x} \int_{y_1}^{w_0} \cdots \int_{y_n}^{w_{n-1}} f^{(n+1)}(w_n) dw_n \cdots dw_0 \qquad (1)$$

$$E(x) = \underbrace{E(x_1)}_{=0} + \int_{x_1}^{x} E'(w_0)\, dw_0$$

let $E'(z) = 0$

$$E(x) = \int_{x_1}^{x} \Big(\underbrace{E'(z)}_{=0} + \int_{z}^{w_0} E''(w_1)\, dw_1 \Big) dw_0$$

$x \le x_1 + h$

$$|E(x)| \le \int_{x_1}^{x} \cdots \int |f^{(n+1)}(\xi)|\, dw_n \cdots dw_0 \le \frac{|f^{(n+1)}(\xi)|}{n!} h^n$$

Making Use of Interpolants

Suppose we can approximate a function as a polynomial:

$$f(x) \approx a_0 + a_1 x + a_2 x^2 + a_3 x^3.$$

How is that useful? E.g. what if we want the integral of f?

$$\int_s^t f(x)\,dx \approx a_0(t-s) + \frac{a_1}{2}(t-s)^2 + \dots$$

$$\approx \sum_{i=0}^{n} \frac{a_i}{(i+1)!}(t-s)^{i+1}$$

Demo: Computing π with Interpolation

More General Functions

Is this technique limited to the monomials $\{1, x, x^2, x^3, \dots\}$?

No, not at all. Works for any set of functions $\{\varphi_1, \dots, \varphi_n\}$ for which the generalized Vandermonde matrix

$$\begin{pmatrix} \varphi_1(x_1) & \varphi_2(x_1) & \cdots & \varphi_n(x_1) \\ \varphi_1(x_2) & \varphi_2(x_2) & \cdots & \varphi_n(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_1(x_n) & \varphi_2(x_n) & \cdots & \varphi_n(x_n) \end{pmatrix}$$

is invertible.

Interpolation with General Sets of Functions

For a general set of functions $\{\varphi_1, \ldots, \varphi_n\}$, solve the linear system with the generalized Vandermonde matrix for the coefficients $(a_1, \ldots, a_n)$:

$$\underbrace{\begin{pmatrix} \varphi_1(x_1) & \varphi_2(x_1) & \cdots & \varphi_n(x_1) \\ \varphi_1(x_2) & \varphi_2(x_2) & \cdots & \varphi_n(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_1(x_n) & \varphi_2(x_n) & \cdots & \varphi_n(x_n) \end{pmatrix}}_{V} \underbrace{\begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix}}_{\boldsymbol{a}} = \underbrace{\begin{pmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_n) \end{pmatrix}}_{\boldsymbol{f}}.$$

> Given those coefficients, what is the interpolant $\tilde{f}$ satisfying $\tilde{f}(x_i) = f(x_i)$?

$$\tilde{f}(x) = \varphi_1(x) \cdot a_1 + \varphi_2(x) \cdot a_2 + \ldots$$

$$= \sum_{i=1}^{n} a_i \varphi_i(x)$$

In-class activity: Interpolation

Outline

Randomness: Why?

What types of problems can we solve with the help of random numbers?

We can compute (potentially) *complicated averages*.

- Where does 'the average' web surfer end up? (PageRank)
- How much is my stock portfolio/option going to be worth?
- How will my robot behave if there is measurement error?

Random Variables

What is a random variable?

A random variable X is a function that depends on 'the (random) state of the world'.

Example: X could be

- 'how much rain tomorrow?', or
- 'will my buttered bread land face-down?'

Idea: Since I don't know the entire state of the world (i.e. all the influencing factors), I can't know the value of X.

$\rightarrow$ Next best thing: Say something about the *average* case.

To do that, I need to know how likely each individual value of X is. I need a probability distribution.

Probability Distributions

What kinds of probability distributions are there?

Continuous $p(x)$ for $x \in [a,b]$

$$\int_a^b p(x)\,dx = 1 \qquad p(x) \geq 0$$

Discrete $X = x_1$ $X = x_2$ ... $X = x_n$

p_1 p_2 p_n

$$\sum_{i=1}^{n} p_i = 1 \qquad p_i > 0$$

Probability $X = x_i$ is 0, but $\int_{x_i}^{x_j} p(x) \geq 0$

Demo: Plotting Distributions with Histograms

Expected Values/Averages: What?

Define '**expected value**' of a random variable.

$$E[f(x)] = \int p(x) \cdot f(x)\, dx \overset{\text{disc.}}{=} \sum_{i=1} p_i\, f(x_i)$$

Define **variance** of a random variable.

$$\sigma^2[X] = E[(X - E[X])^2]$$
$$= E[X^2] - E[X]^2$$

Expected Value: Example I

What is the expected snowfall in Champaign?

$$E[\text{Snow}] = \int_{\text{day} \in \text{year}} \text{Snow}(\text{day}) \cdot p(\text{day}) \, d\text{day}$$

Snow - Random Variable

amount of snow in 1 year

Tool: Law of Large Numbers

Terminology:

- *Sample*: A sample $s_1, \ldots, s_N$ of a discrete random variable X (with potential values $x_1, \ldots, x_n$) selects each s_i such that $s_i = x_j$ with probability $p(x_j)$.

In words:

- As the number of samples $N \to \infty$, the average of samples converges to the expected value with probability 1.

> What can samples tell us about the distribution?

$$E[X] = \lim_{N \to \infty} \frac{1}{N} \left(\sum_{i=1}^{N} s_i \right)$$

$$= \sum_{i=1}^{n} x_i \cdot p(x_i)$$

Sampling: Approximating Expected Values

Integrals and sums in expected values are often challenging to evaluate.

How can we approximate an expected value?
Idea: Draw random samples. Make sure they are distributed according to $p(x)$.

$$E[f(x)] \approx \frac{1}{N}\sum_{i=1}^{N} f(s_i)$$

What is a Monte Carlo (MC) method?

MC method computes an approximation based on a random sample.

Expected Values with Hard-to-Sample Distributions

Computing the sample mean requires samples from the distribution $p(x)$ of the random variable X. What if such samples aren't available?

$$E[X] = \int x \cdot p(x)\, dx$$

$$= \int x \cdot \frac{p(x)}{\tilde{p}(x)} \cdot \tilde{p}(x)\, dx$$

$\tilde{p}(x)$ that is easy to sample from

Switching Distributions for Sampling

Found:

$$E[X] = E\left[\tilde{X} \cdot \frac{p(\tilde{X})}{\tilde{p}(\tilde{X})}\right]$$

Why is this useful for sampling?

$\tilde{s}_1 \ldots \tilde{s}_N$ from $\tilde{X}$, obeying $\tilde{p}(\tilde{x})$

$$E[X] \approx \frac{1}{N}\sum_{i=1}^{N} \tilde{s}_i \cdot \frac{p(s_i)}{\tilde{p}(s_i)}$$

Expected Value: Example II

What is the expected snowfall in Illinois?

Dealing with Unknown Scaling

What if a distribution function is only known up to a constant factor, e.g.

$$p(x) = C \cdot \underbrace{\begin{cases} 1 & \text{point } x \text{ is in IL,} \\ 0 & \text{it isn't.} \end{cases}}_{q(x)}$$

Typically $\int_{\mathbb{R}} q \neq 1$. We need to find C so that $\int p = 1$, i.e.

$$C = \frac{1}{\int_{\mathbb{R}} q(x)dx}.$$

Idea: Use sampling.

Demo: Computing π using Sampling
Demo: Errors in Sampling

Sampling: Error

The Central Limit Theorem states that with

$$S_n := s_1 + s_2 + \cdots + s_n$$

for the (s_i) independent and identically distributed according to random variable X with variance σ^2, we have that

$$\frac{S_n - nE[X]}{\sqrt{\sigma^2 n}} \to \mathcal{N}(0, 1),$$

i.e. that term approaches the normal distribution. Or, short and imprecise,

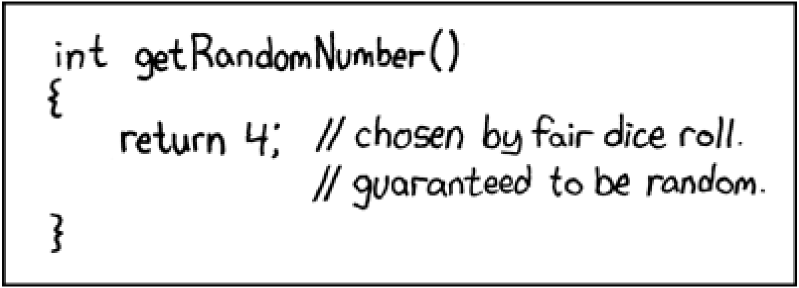
$$\left| \frac{1}{n} S_n - E[X] \right| = O\left(\frac{1}{\sqrt{n}} \right).$$

Monte Carlo Methods: The Good and the Bad

What are some *advantages of MC methods?*

What are some *disadvantages* of MC methods?

Computers and Random Numbers



[from xkcd]

How can a computer make random numbers?

Random Numbers: What do we want?

What properties can 'random numbers' have?

- Have a specific distribution (often 'uniform'–each value between, say, 0 and 1, is equally likely)
- Real-valued/integer-valued
- Repeatable (i.e. you may *ask* to exactly reproduce a sequence)
- Unpredictable
  - V1: 'I have no idea what it's going to do next.'
  - V2: No amount of engineering effort can get me the next number.
- Uncorrelated with later parts of the sequence (Weaker: Doesn't repeat after a short time)
- Usable on parallel computers

What's a Pseudorandom Number?

Actual randomness seems like a lot of work. How about 'pseudo-random numbers?'

Idea: Maintain some 'state'. Every time someone asks for a number:

$$\text{random\_number}, \text{new\_state} = f(\text{state})$$

Satisfy:

- Distribution
- 'I have no idea what it's going to do next.'
- Repeatable (just save the state)
- Typically *not* easy to use on parallel computers

Demo: Playing around with Random Number Generators

Some Pseudorandom Number Generators

Lots of variants of this idea:

- LC: 'Linear congruential' generators
- MT: 'Mersenne twister'

Remarks:

- Initial state and parameter choice often surprisingly tricky.
  Bad choice: Predictable/correlated numbers.
  E.g. Debian OpenSSL RNG disaster
- Absolutely **no reason** to use LC or MT any more. (Although almost all randonumber generators you're likely to find are based on those–Python's `random` module, `numpy.random`, C's `rand()`, C's `rand48()`.
- These are **obsolete.**

Counter-Based Random Number Generation (CBRNG)

What's a CBRNG?

Idea: Cryptography has *way* stronger requirements than RNGs.
And the output *must* 'look random'.

E.g. AES:
128 encrypted bits $= \mathrm{AES}\,(\text{128-bit plaintext}, \text{128 bit key})$

Read that as:
128 random bits $= \mathrm{AES}\,(\text{128-bit counter}, \text{arbitrary 128 bit key})$

- Just use $1, 2, 3, 4, 5, \ldots.$ as the counter.
- *No* quality requirements on counter or key to obtain high-quality random numbers
- *Very* easy to use on parallel computers
- Often accelerated by hardware, faster than the competition

Demo: Counter-Based Random Number Generation

Outline