Week #6 : Integrals - Modeling

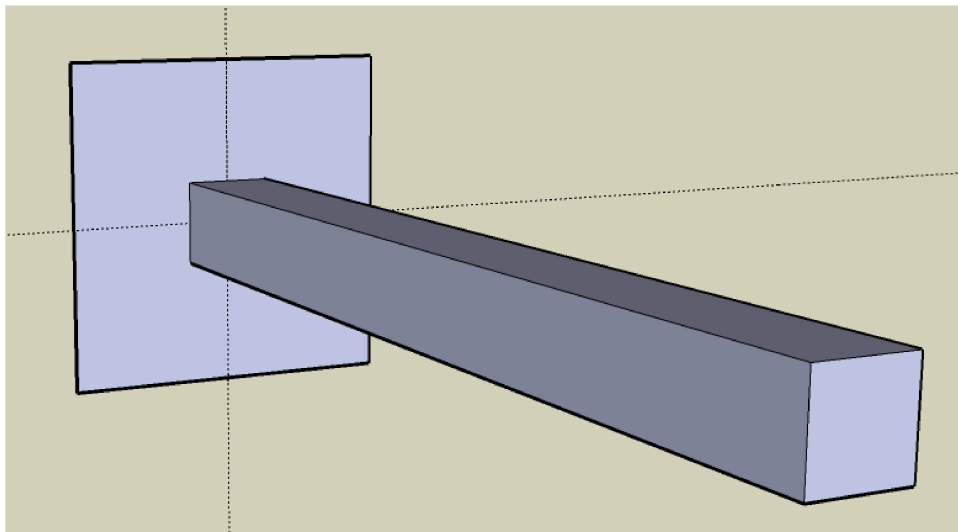
Goals:

- Use MATLAB to solve a variety of integration problems.
- Use integration to find the average value of a function.
- Use MATLAB to find the average value of a function.
- Use MATLAB to find the average value of a sequence of data.

Numerical Integration - Motivation

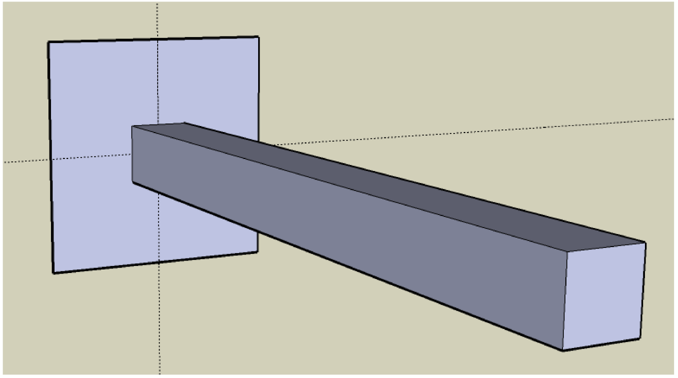
A common engineering challenge is to transfer heat generated by a motor or combustion into a nearby fluid.

This transfer is often made more effective by the use of **cooling fins**, which increase the surface area of contact with the fluid.

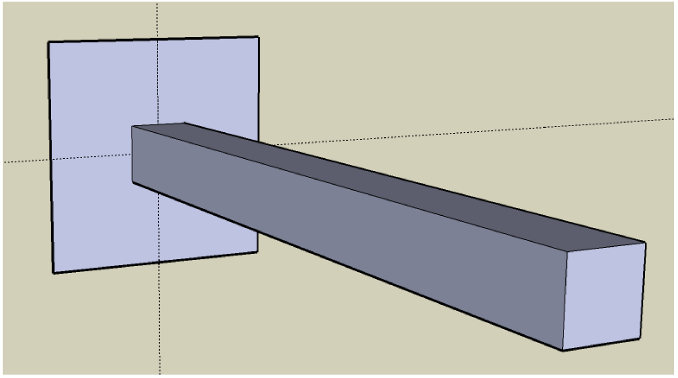


If we fix the temperature at the base, we can ask the question **"How quickly is heat radiated out of fin?"**

Problem. What factors affect the rate of heat transfer out of the fin?



Problem. Take a small slice of length Δx of the fin: how much heat is lost through that slice?



Give an expression for the total amount of heat lost over the whole fin.

Integration

As soon as you see any sum of the form $\sum \ldots \Delta x$, you should be thinking "Integral"!

For our fin example, net heat flow to environment is

If, for simplicity, we assume $T_\infty = 0$, our target integral becomes

Finding the Temperature Distribution

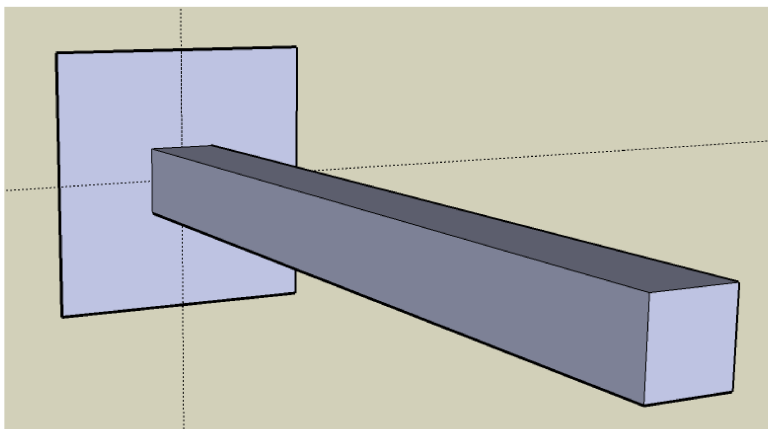
To evaluate the integral, we first need to find the temperature distribution along the fin. Without getting into all the gory details, the temperature along the fin will be given by the formula

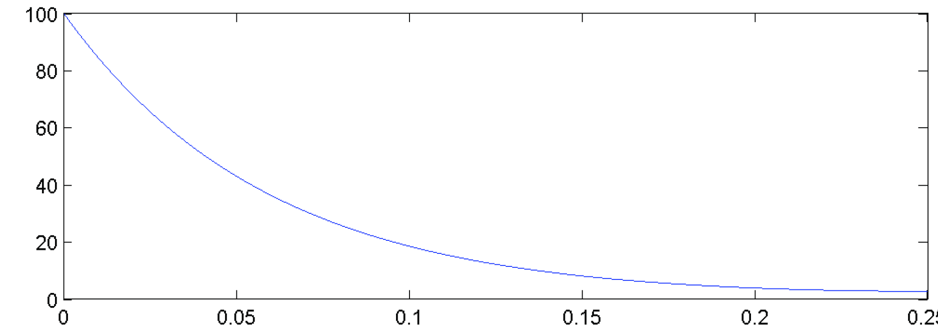
$$T(x) = \frac{T_b\left(\cosh(m(L-x)) + \frac{h}{mk}\right)(\sinh(m(L-x)))}{\cosh(mL) + \frac{h}{mk}\sinh(mL)}$$

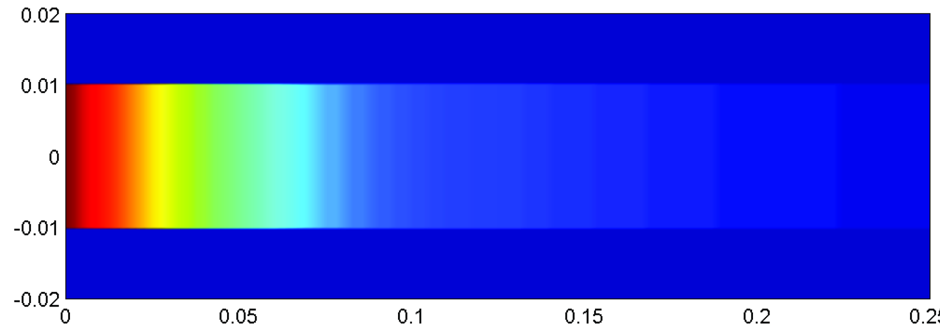
Problem. What are these new $\cosh(x)$ and $\sinh(x)$ functions?

Graphically

Once we have the temperature distribution, we can graph it along the length of the fin to see if it makes sense.







Computing Total Heat Loss

Now, we have addressed one challenge in our problem: we know the steady-state temperature along the fin. Next, we want to compute the net rate of heat flow out, or the cooling ability of the fin.

The heat flow out of the fin is given by

$$Q = \int_0^L h \; P \; T(x) \; dx \qquad (T_\infty \text{ assumed} = 0)$$

Our first approach, if it is possible, should be direct anti-differentiation (think $\int x^2 \; dx = \frac{1}{3}x^3$).

For this problem, given the earlier temperature we found, $T(x)$, we **can** evaluate the integral exactly:

$$Q = \frac{1}{2}\frac{T_b \, P \, h \, (m \, k \, exp(2 \, m \, L) - m \, k + h \, exp(2 \, m \, L) + h - 2 \, h \, exp(m \, L)}{m(\cosh(m \, L) \, m \, k + h \; \sinh(m \, L)) \; exp(-m \, L)}$$

Comments on Anti-Derivatives

Through this last step, we reached what would be the important engineering goal: obtaining the **numerical value** for the integral. When we compute integrals analytically, by using anti-derivatives, we are doing the best possible thing.

- Integrals give exact values.
- Integrals can be re-used immediately with different constants.

Unfortunately, actually computing the numerical value of an integral using antiderivatives isn't always an option:

- Some functions don't have antiderivatives.
- Sometimes we don't have a function, but only data.
- Sometimes we forget how to find the anti-derivative!

Numerical Quadrature

The word *quadrature* comes from the Greek challenge of trying to *square the circle*, or finding the **area** (in square units) of the round circle.

When you hear **quadrature** think **numerical integration**.

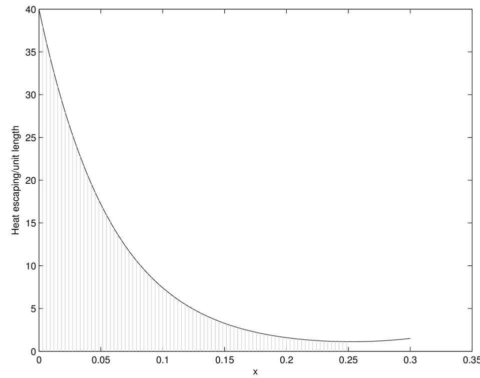
The two common scenarios where we need *numerical* integration will be:

- **Formula** for $f(x)$ known, want $\displaystyle\int_a^b f(x)dx$
- **Data** for $f(x)$ collected at $f(x_i)$, want $\displaystyle\int_a^b f(x)dx$

We will study the formula case initially, because it is easier to experiment with. We will continue to use our temperature example,

$$\int_0^L h\ P\ T(x)\ dx = 2.36269950112023 \text{ J/s}$$

Graphically, $\displaystyle\int_0^L \underbrace{h\ P\ T(x)}_{f(x)}\ dx$ is the area shown below:



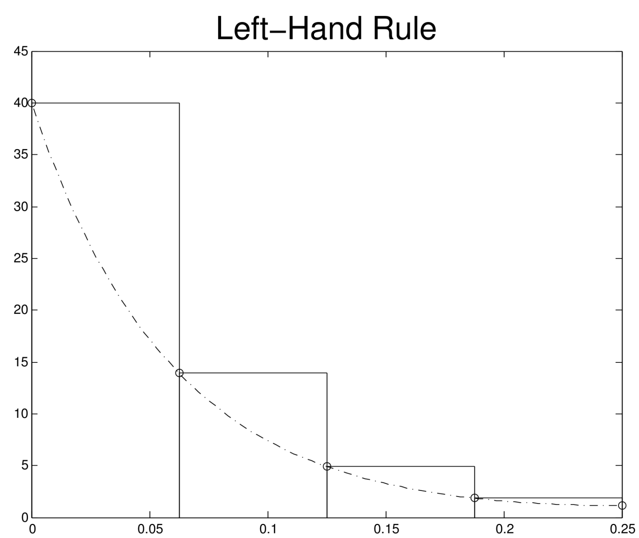
Numerical integration is performed by

- separating the desired interval into *panels*
- on each *panel*, evaluating the integrand, $f(x)$ one or more times and those values are combined in some way to *estimate* the area of the panel

Left-Hand Sum

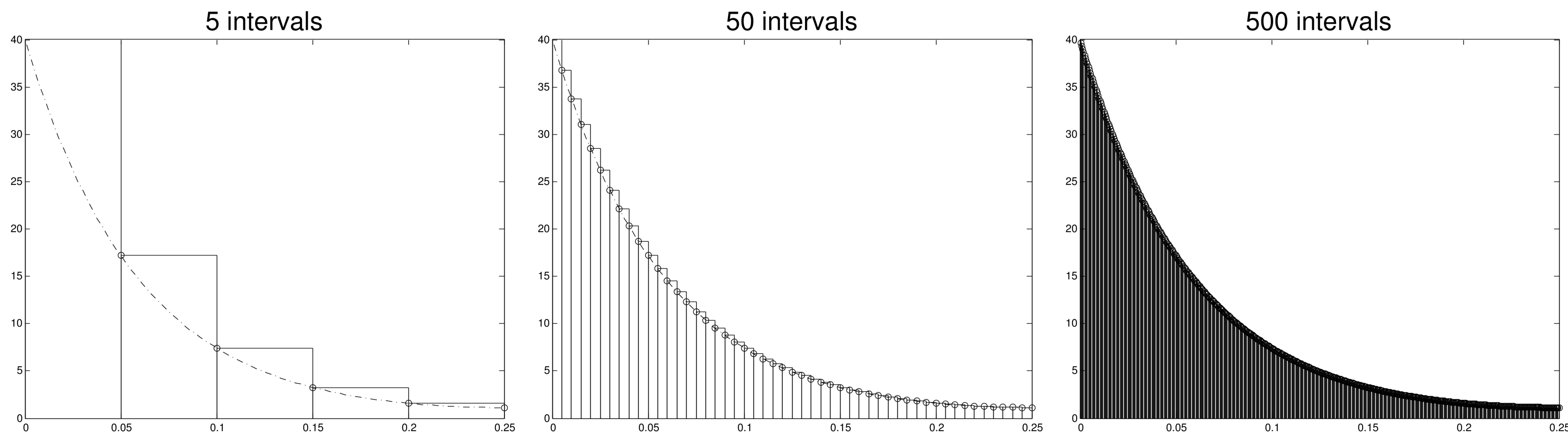
The simplest quadrature rule is one we have already seen: the LEFT(n) sum.

- Divide the interval into n panels, width $\Delta x = (b-a)/n$
- Evaluate function at the left end point, $f(x_{i-1})$ on each panel.
- Compute the area of rectangles, $\displaystyle\sum_{i=1}^{N} f(x_{i-1}) \cdot \Delta x$ or



Quadrature Principles

We are approximating a complex shape with simpler shapes for which we can compute the area. The **more panels** we use, the **more accurate** the area estimate will be:



By using enough panels, we can reduce the error to any level we like, but then it takes longer to compute.

Our real question should be: as we do **more work**, how much **more accurate** does our answer become?

Problem. Download the file `W7_1.m`, and extend it so it plots the graph of the temperature along the fin.

Problem. Estimate the integral $Q = \int_0^L hP\ T(x)\ dx$ with the left-hand rule,

$$Q \approx \sum_{i=1}^{n} hP\ T(x_i)\ \Delta x$$

Problem. Repeat the calculation, but using the **`sum`** command instead of a loop. Loops tend to be slow in MATLAB, compared to built-in matrix operations.

Problem. Add a print statement that shows the number of intervals used, and the resulting error. Experiment by doubling the number of intervals and seeing the resultant reduction in error.

Problem. Doubling the number of panels does what to the error?

We now have a procedure for estimating integrals **without** needing to actually use any calculus. Our next quest will be to make these integral estimates as *accurate* as possible for the *least amount of work*. In other words, we will be looking for more **efficient** methods.