A discrete-time **signal** is a function of integer times, such as $x[n]$. A discrete-time **system** converts input signals to output signals according to a fixed rule. In 6.01, we focus almost exclusively on the class of systems that are *linear* and *time-invariant* (LTI). This is highly restrictive, but it happens that a great many systems in the real world can be approximated as being LTI.

We have three perspectives from which we can look at and manipulate LTI systems: difference equations, transfer functions, and block diagrams. Each has its strengths and weaknesses, and a large part of your effort will be spent either in deciding which viewpoint to take or in converting between them. This is a general feature of signal processing and systems theory: placing a problem in the correct basis oftentimes goes a long way towards solving it.

1 Difference Equations

1. Any LTI system relating input $x[n]$ to output $y[n]$ can be described uniquely by means of a difference equation:

$$a_0x[n] + a_1x[n-1] + \ldots + a_Nx[n-N] = b_0y[n] + b_1y[n-1] + \ldots + b_Ny[n-N] \qquad (1)$$

2. **Good for:** numerically finding an output $y[n]$ point-by-point given initial conditions and an input waveform $x[n]$. We can always solve for the output at time n depending on its previous values and the current and previous values of the input:

$$y[n] = \left(\frac{1}{b_0}\right)(a_0x[n] + a_1x[n-1] + \ldots + a_Nx[n-N] - b_1y[n-1] - \ldots - b_Ny[n-N]) \quad (2)$$

3. **Good for:** Python implementation. We basically copied the above equation into Python each time we implemented control loops (robot down the hallway, head, searching for light, etc).

4. **Combining Difference Equations:** Difference equations can *sometimes* be easily combined by means of algebra.
Example 1: Say relation 1 relates input $x[n]$ and intermediate signal $z[n]$ by

$$z[n] = x[n] - x[n-1]$$

and relation 2 relates $z[n]$, $x[n]$, and output $y[n]$ by

$$y[n] = z[n-1] + x[n-2]$$

Can we find a relation between $y[n]$ and $x[n]$? If we delay equation by one time unit to obtain

$$z[n-1] = x[n-1] - x[n-2]$$

This can be plugged into equation 2 to yield the desired relation:

$$y[n] = x[n-1]$$

Example 2: Now suppose equation 1 was instead

$$z[n] - z[n-1] = x[n] - x[n-2]$$

and equation 2 was

$$y[n] = z[n-2] + z[n]$$

One may still combine them by summing appropriately delayed versions of each equation, but it is a painful procedure. Transfer functions are far better suited to this task.

2 Transfer Functions

As we use them in 6.01, transfer functions are a convenient means for us to keep track of the delays and coefficients of a difference equation. In reality, there is an incredibly deep and beautiful theory behind them (and systems in general) — we advise interested parties to check out 6.003 (or the OCW site if you can't wait!).

A transfer function is a rational polynomial of a single "variable," $\mathcal{R}$, that specifies the difference equation relating our input $x[n]$ and output $y[n]$. Suppose an LTI system is described by the generic difference equation we saw earlier:

$$a_0x[n] + a_1x[n-1] + \ldots + a_Nx[n-N] = b_0y[n] + b_1y[n-1] + \ldots + b_Ny[n-N].$$

We can construct a transfer function from this description.

1. Replace all terms $ax[n-M]$ with $aX\mathcal{R}^M$. The variable $\mathcal{R}$ represents a delay, and X is the transfer function variable for the input signal.
2. Do likewise for all terms $by[n-M]$.
3. Solve for the ratio Y/X in terms of $\mathcal{R}$. This ratio is the transfer function.

One may reverse these steps to obtain a difference equation from a transfer function.
Several important notes about transfer functions deserve mentioning:

1. **Good for:** Analyzing behavior at a high level (stable? oscillatory?).
2. **Good for:** Combining systems.
3. **Combining Transfer Functions: Composition** One may compose transfer functions as if they are "ratios" between signals. For instance, if $H_1 = Z/X$ is the transfer function from X to Z and $H_2 = Y/Z$ is from Z to Y, the two may be **multiplied** to yield the transfer function from X to Y: $H_1 \cdot H_2 = Y/X$.
4. **Combining Transfer Functions: Addition** Continuing the ratio approach, if $H_1 = Z_1/X$ is the transfer function from X to Z_1, $H_2 = Z_2/X$ is from X to Z_2, and $Y = Z_1 + Z_2$, then the transfer function from X to Y is given by $Y/X = (Z_1 + Z_2)/X = H_1 + H_2$.
5. **Remember:** You can treat transfer functions as if they are fractions. If the numerator possesses a factor also present in the denominator, they can be canceled out without changing the system.
6. **Remember:** Transfer functions are rational polynomials in terms of $\mathcal{R}$. If yours isn't (e.g. if X or Y is showing up in it), a red flag should go off that something is amiss.
7. **Example** Consider the problem we "gave up" on in the previous section (difference equations). We are given the relationship between $z[n]$ and $x[n]$ as $z[n] - z[n-1] = x[n] - x[n-2]$, and we are also given a relationship between $z[n]$ and $y[n]$ as $y[n] = z[n-2] + z[n]$. What is the difference equation between $y[n]$ and $x[n]$? We can go through this step-by-step.
   - Converting equation 1 to transfer function form: $Z(1-\mathcal{R}) = X(1-\mathcal{R}^2)$
   - Converting equation 2 to transfer function form: $Y = Z(1+\mathcal{R}^2)$
   - Solve for Z in equation 1: $Z = X(1-\mathcal{R}^2)/(1-\mathcal{R})$
   - Plug into equation 2: $Y/X = (1-\mathcal{R}^2)(1+\mathcal{R}^2)/(1-\mathcal{R})$
   - Simplifying: $Y/X = (1+\mathcal{R})(1+\mathcal{R}^2)$
   - Expanding: $Y = X(\mathcal{R}^3 + \mathcal{R}^2 + \mathcal{R} + 1)$
   - Convert to the difference equation picture: $y[n] = x[n-3] + x[n-2] + x[n-1] + x[n]$

3 Block Diagrams

Block diagrams are graphical representations of LTI systems. In 6.01, we consider block diagrams made from three basic pieces.

1. **Gain.** The multiplication of a signal $x[n]$ with a constant gain K is produced at the output of a gain block, $Kx[n]$.

2. **Delay.** A signal is delayed by one time unit before being produced at the output.

3. **Summation.** The sum of two or more signals is produced at the output of summation block.

Every block is fully characterized by a transfer function relating its inputs to its output. We can therefore replace a complicated block network with a single block, only keeping the transfer function $H = Y/X$ of the more complicated system.

1. **Good for:** Linking a physical description of a system to a mathematical one.
2. **Good for:** Circuit implementations. We see a direct correlation between two of the basic blocks (summation and gain) and circuits (Summer/Subtractor, Amplifier).

3. **Remember:** The lines connecting the blocks each correspond to a signal, much as a voltage corresponds to a node of a circuit. When a wire "splits" into multiple threads, they all still contain the same signal

4. **Remember:** Block diagrams are not unique. A system can be described with any of an infinite number of block diagrams. Just because two diagrams aren't exactly the same does not mean that they have different input/output relationships.

4 Going Between the Three

Conversion between difference equations and transfer functions is a relatively mechanistic process. It is less clear how to obtain a block diagram from a difference equation/transfer function, or vice-versa, so we focus on that.

4.1 Difference Equation to Block Diagram

This is in many ways a design problem: how to implement difference equations in terms of discrete gain/sum/delay blocks? There are many possible answers.

Example: If the system is specified in terms of a single difference equation, of the form

$$a_0x[n] + a_1x[n-1] + \ldots + a_Nx[n-N] = b_0y[n] + b_1y[n-1] + \ldots + b_My[n-M]$$

then certain "canonical" forms allow us to generate a block diagram almost automatically. For instance, we may rearrange the difference equation to solve for $y[n]$:

$$y[n] = \left(\frac{1}{b_0}\right)(a_0x[n] + a_1x[n-1] + \ldots + a_Nx[n-N] - b_1y[n-1] - \ldots - b_My[n-M])$$

Can you implement the above with a block diagram? Think of it as a two-step process: assume, first that you have access to the signals $x[n], x[n-1], \ldots$ and $y[n-1], y[n-2], \ldots$; can you use these signals to generate $y[n]$? Then, think about how you might generate these signals.

The above technique can always be used to convert a single difference equation into a block diagram. This is not by itself the most convenient form, however, when multiple equations and intermediate signals are involved. One approach is to reduce the multiple equations into a single equation by means of transfer functions, and then implement this single equation in the manner we just did. The other approach is to be creative: with practice you will "see" ways to rapidly implement difference relationships.

4.2 Block Diagram to Transfer Function

Obtaining a difference equation from a block diagram is analogous to analytically solving a circuit. The brute force approach is to label every node in the diagram, write the transfer relationship for every block, and combine the resulting set of equations into a single transfer function from input to output. You might find yourself skipping steps after a little practice with this; oftentimes you'll be able to go straight from the block diagram to a transfer function.

Example: Consider the standard feedback block diagram: Y is passed through a transfer function H_2, the resulting signal is subtracted from X to give an error signal, and this error signal is sent through a different transfer function H_1 to determine Y.

Passing Y through H_2 creates the signal

$$V = H_2 Y$$

This is then subtracted from X to give us an error signal, E.

$$E = X - V$$

The next block, H_1, turns E into Y, giving us the equation

$$Y = H_1 E$$

From this point, we can combine these equations algebraically. (1) and (2) fuse together to give the relationship between E, X, and Y: $E = X - H_2 Y$. Inserting this into the second equation gives us a transfer relationship between X and Y:

$$Y = H_1 X - H_1 H_2 Y$$

Solving for Y, we have the global transfer relationship between X and Y — otherwise known as Black's Formula.

$$Y = \frac{H_1}{1 + H_1 H_2} X$$