

# Tips and Tricks - Combining Functions Using Anonymous Functions

By [Loren Shure](#)

Anonymous functions let you create simple functions as variables without having to store the functions in a file. You can construct complex expressions by combining multiple anonymous functions. Here are some sample combinations.

## Function Composition

In this example we will create a single function by having one function call another.

Suppose we want to compute the standard deviation of the mean of some data—for example, `std(mean(x))`. We can construct separate anonymous functions to compute the mean and standard deviation and then combine them:

```
f = @mean;
g = @std;
fCompG = @(x) g(f(x));
```

To ensure that the function composition works as expected, we evaluate the first function and use its output as input to the second. We then check that this two-step process is equivalent to the one-step evaluation of the result that we obtained from function composition:

```
x = rand(10,4);
fx = f(x);
gfx = g(fx);
gfx2 = fCompG(x);
gfsEqual = isequal(gfx,gfx2)
gfsEqual = 1
```

We could create a generalized composition operator  $g(f)$  and use that instead:

```
compose = @(g,f)@(x)g(f(x))
fCompG2 = compose(g,f)
gfcx = fCompG2(x);
gfcEqual = isequal(gfx, gfcx)
gfcEqual = 1
```

## Conditional Function Composition

Now let's look at a more complicated composition. Suppose that we want to compute the expression:

$y = \sin(x) - \text{mean}(x) + 3$ , but we don't always want to subtract the mean or add 3. We dynamically build one function to compute  $y$ , which subtracts the mean or adds 3 only when we want it to.

Let's start by building function handles for computing the sine, subtracting the mean, and adding 3:

```
x = 0:pi/100:pi/2;
myfunc = @sin;
meanValue = @mean;
three = @(x) 3;
```

While the variable containing each function handle retains its name, the function it describes can change. Note that this example works only when  $x$  is a row or column vector. For multidimensional data, the expressions would be more complicated, and we would use `bsxfun`.

We combine these functions conditionally. For simplicity, let's say we want to add 3 but don't want to subtract the mean. In a more realistic example, we would use logical comparisons to determine which functions to apply:

```
if false
myfunc = @(x) myfunc(x) - meanValue(x);
end
if true
myfunc = @(x) myfunc(x) + three(x);
end
```

Let's try this on the data  $x$  that we created earlier:

```
mf3mean = myfunc(x);
```

Whatever functions we combine with the original function, we always have a function handle to evaluate. We can use it to specify additional information as we build up an appropriate function for our calculation.

## A Versatile Technique

Using anonymous functions is a versatile technique that you might find useful in a wide range of calculations, from the simplest to the most complex. We've looked at just two examples. In fact, the number of possibilities is limitless.

### Products Used

- [MATLAB®](#)

### Learn More

- [Loren on the Art of MATLAB](#)

See more articles and subscribe at [mathworks.com/newsletters](http://mathworks.com/newsletters).