Gradient Descent Visualization

Explore how the learning rate parameter affects training in machine learning algorithms

Set up the problem

Here is an example loss function, where we will play with the learning rate of the gradient descent algorithm

```
learningRate = 0.73;
% Rate too low:     0.03
% Rate too high:    10
% Good rate:        0.73
axisCorners = [0 60 0 5];

x = linspace(-5,5,100);
y = x.^2 + 0.3 * randn(size(x));

f = figure;
f.Visible = 'On';
subplot(2,1,1)
plot(x,y,'LineWidth',3)
title('Example Loss function')
xlabel('Weight/bias')
ylabel('Loss')
```

Initialize the algorithm at a specific weight value

```
hold on
a = 22;
x_ = x(a);
stem(x_,y(a), 'LineWidth', 2)
hold off

b = abs(x_);
subplot(2,1,2)
plot(1,b,'LineWidth',2)
axis(axisCorners)
```

Run one step of the gradient descent algorithm

This section can be executed repeatedly to continue the algorithm

```
gradF = gradient(y);
subplot(2,1,1)
hold on
x_ = x_ - learningRate * gradF(a);
a = findClosestIndex(x_,x);
stem(x(a), y(a), 'LineWidth', 2)
hold off

subplot(2,1,2)
b(end+1) = abs(x(a));
plot(1:numel(b),b,'LineWidth',2)
axis(axisCorners)
```

```
title('Loss vs. iteration')
xlabel('Iteration')
ylabel('Loss')
```

Run a whole batch of the gradient descent algorithm

```
for ii = 1:50
    gradF = gradient(y);
    subplot(2,1,1)
    hold on
    x_ = x_ - learningRate * gradF(a);
    a = findClosestIndex(x_,x);
    stem(x(a), y(a), 'LineWidth', 2)
    hold off

    subplot(2,1,2)
    b(end+1) = abs(x(a));
    plot(1:numel(b),b,'LineWidth',2)
    axis(axisCorners)
    title('Loss vs. iteration')
    xlabel('Iteration')
    ylabel('Loss')

    pause(0.2)

end
```

Helper functions

```
function a = findClosestIndex(x_,x)
    [~, a] = min((x_ - x).^2);
end
```