# Gradient Checking

Gradient checking will assure that our backpropagation works as intended. We can approximate the derivative of our cost function with:

$$\frac{\partial}{\partial\Theta}J(\Theta) \approx \frac{J(\Theta+\epsilon)-J(\Theta-\epsilon)}{2\epsilon}$$

With multiple theta matrices, we can approximate the derivative **with respect to $\Theta_J$** as follows:

$$\frac{\partial}{\partial \Theta_J}J(\Theta) \approx \frac{J(\Theta_1,\ldots,\Theta_{J+\epsilon},\ldots,\Theta_n)-J\left((\Theta_1,\ldots,\Theta_{J-\epsilon},\ldots,\Theta_n)\right)}{2\epsilon}$$

A small value for ϵ (epsilon) such as $\epsilon = 10^{-4}$, guarantees that the math works out properly. If the value for ϵ is too small, we can end up with numerical problems.

Hence, we are only adding or subtracting epsilon to the $\Theta_J$ matrix. In octave we can do it as follows:

```
epsilon = 1e-4;
for i = 1:n,
  thetaPlus = theta;
  thetaPlus(i) += epsilon;
  thetaMinus = theta;
  thetaMinus(i) -= epsilon;
  gradApprox(i) = (J(thetaPlus) - J(thetaMinus))/(2*epsilon)
end;

```

We previously saw how to calculate the deltaVector. So once we compute our gradApprox vector, we can check that gradApprox ≈ deltaVector.

Once you have verified **once** that your backpropagation algorithm is correct, you don't need to compute gradApprox again. The code to compute gradApprox can be very slow.