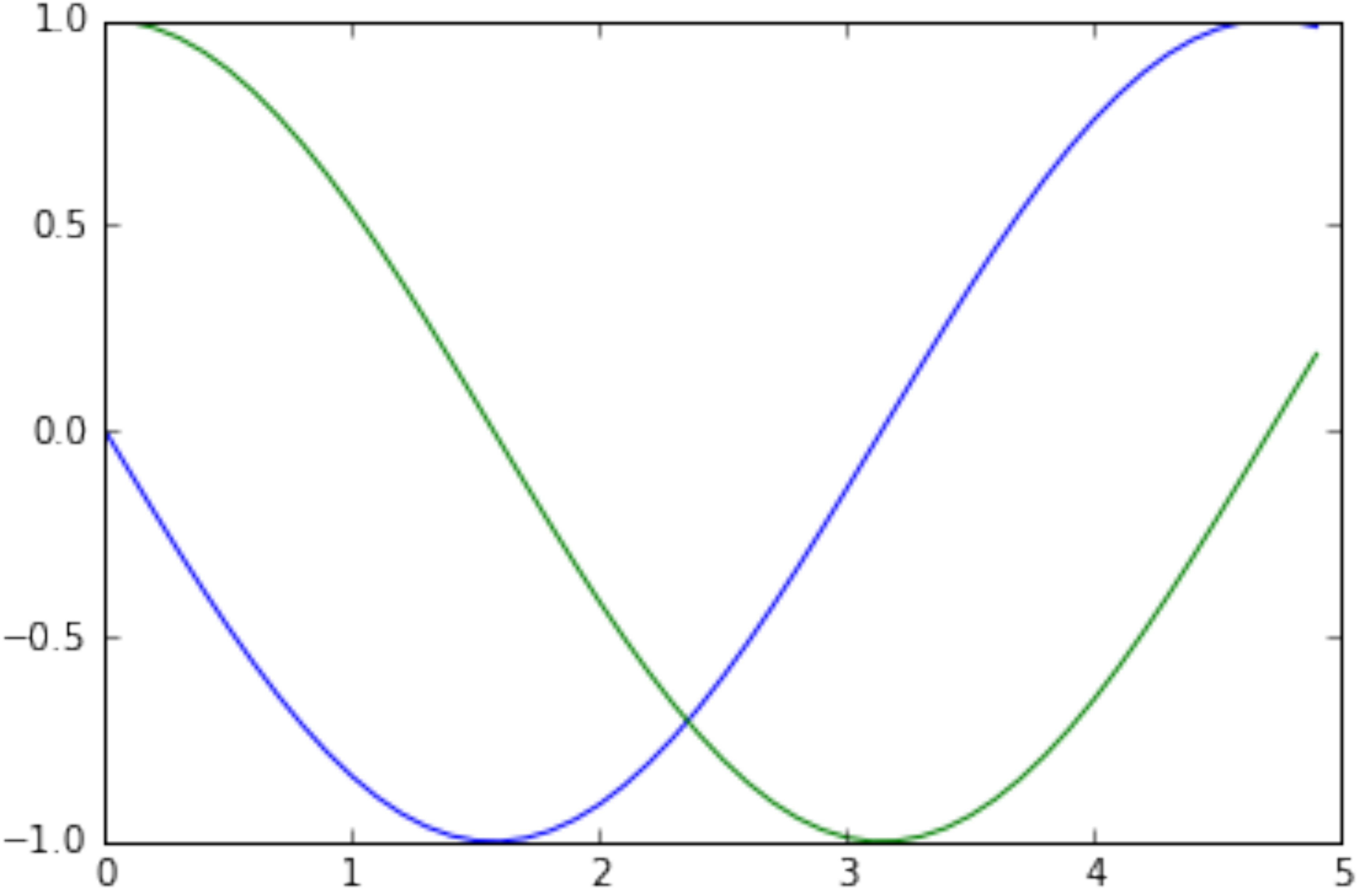
Automatic Differentiation

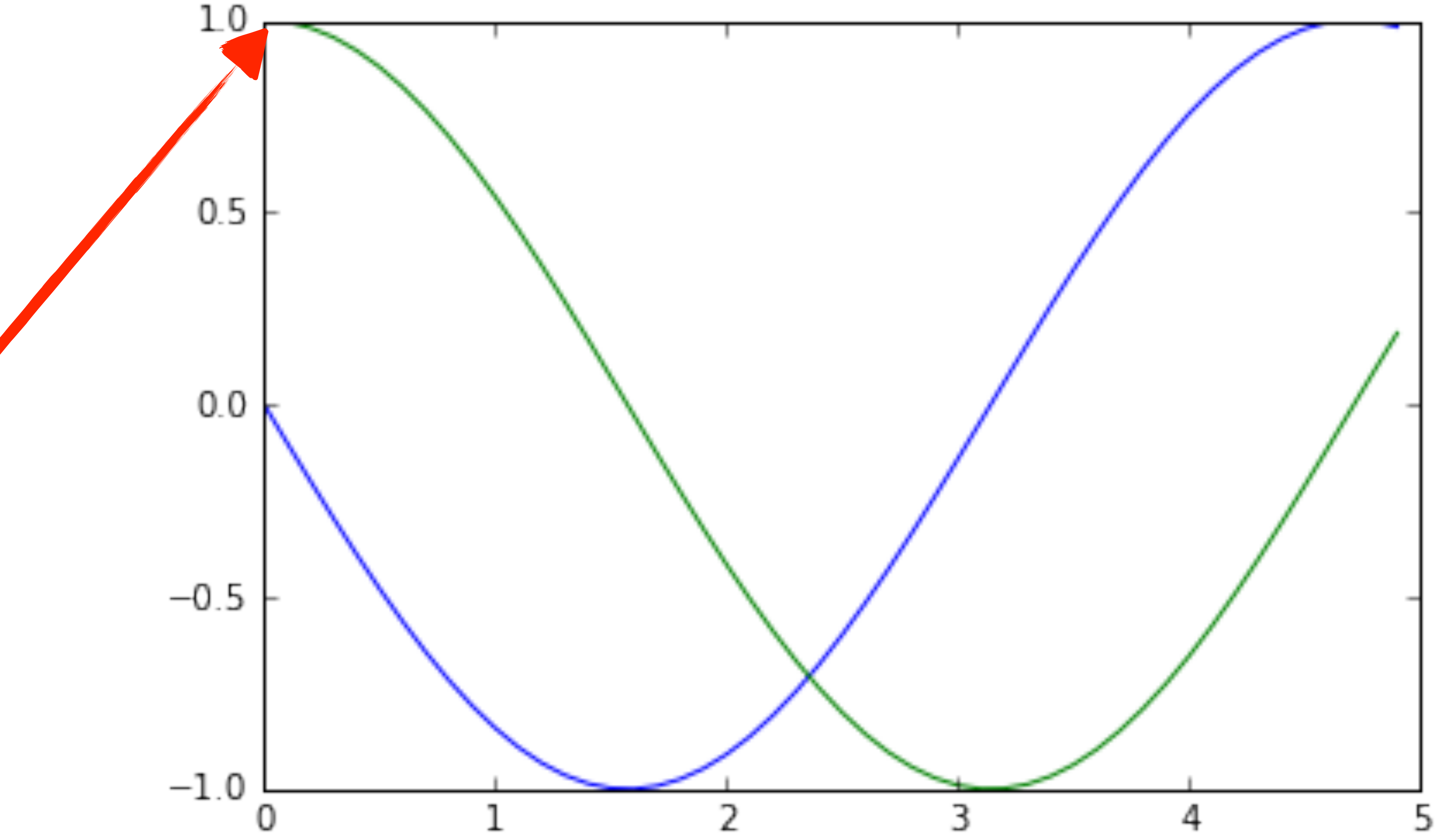
$$cos(x)$$

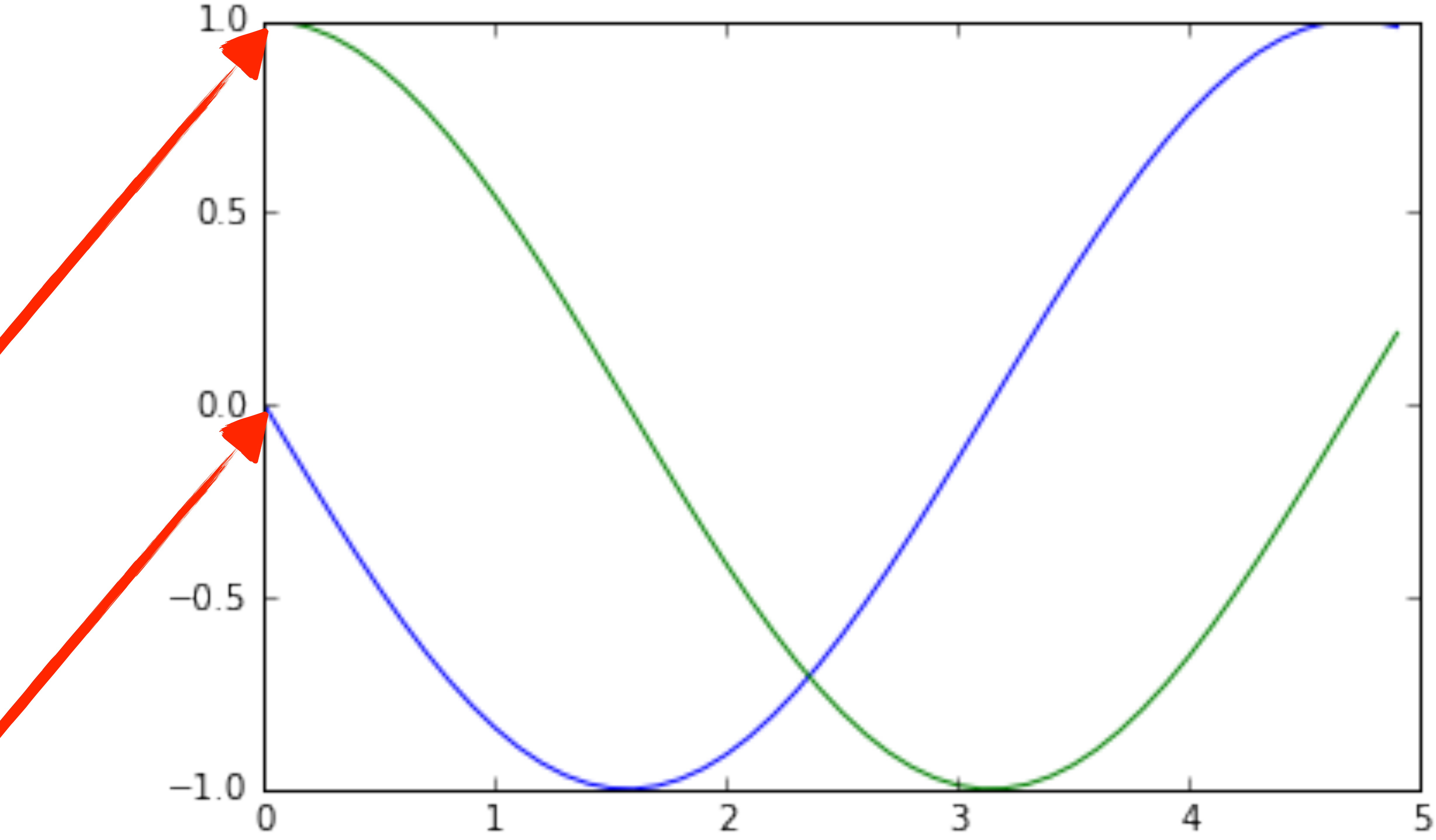
$$cos(x)'$$

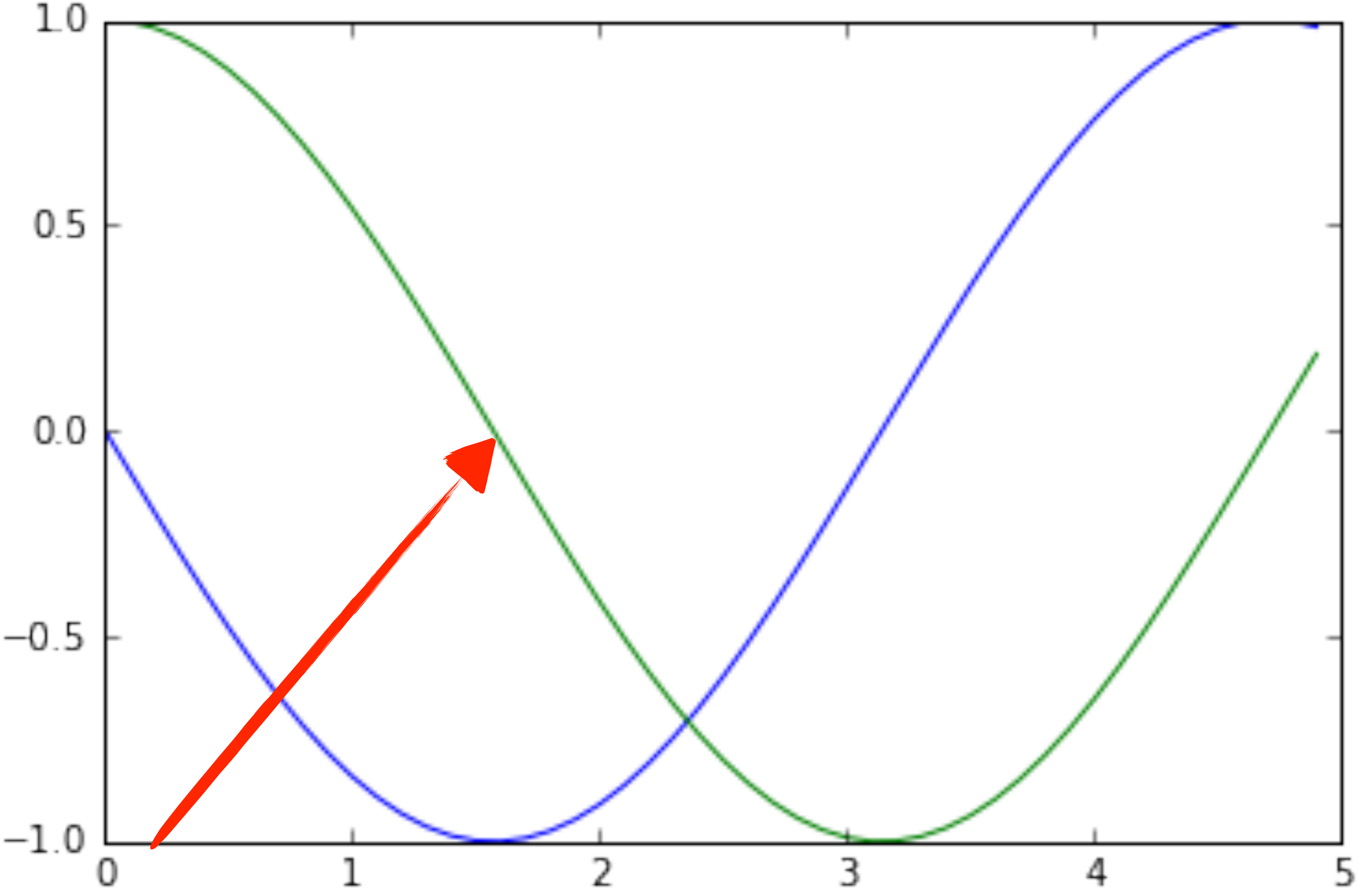
$$cos(x)' = -sin(x)$$

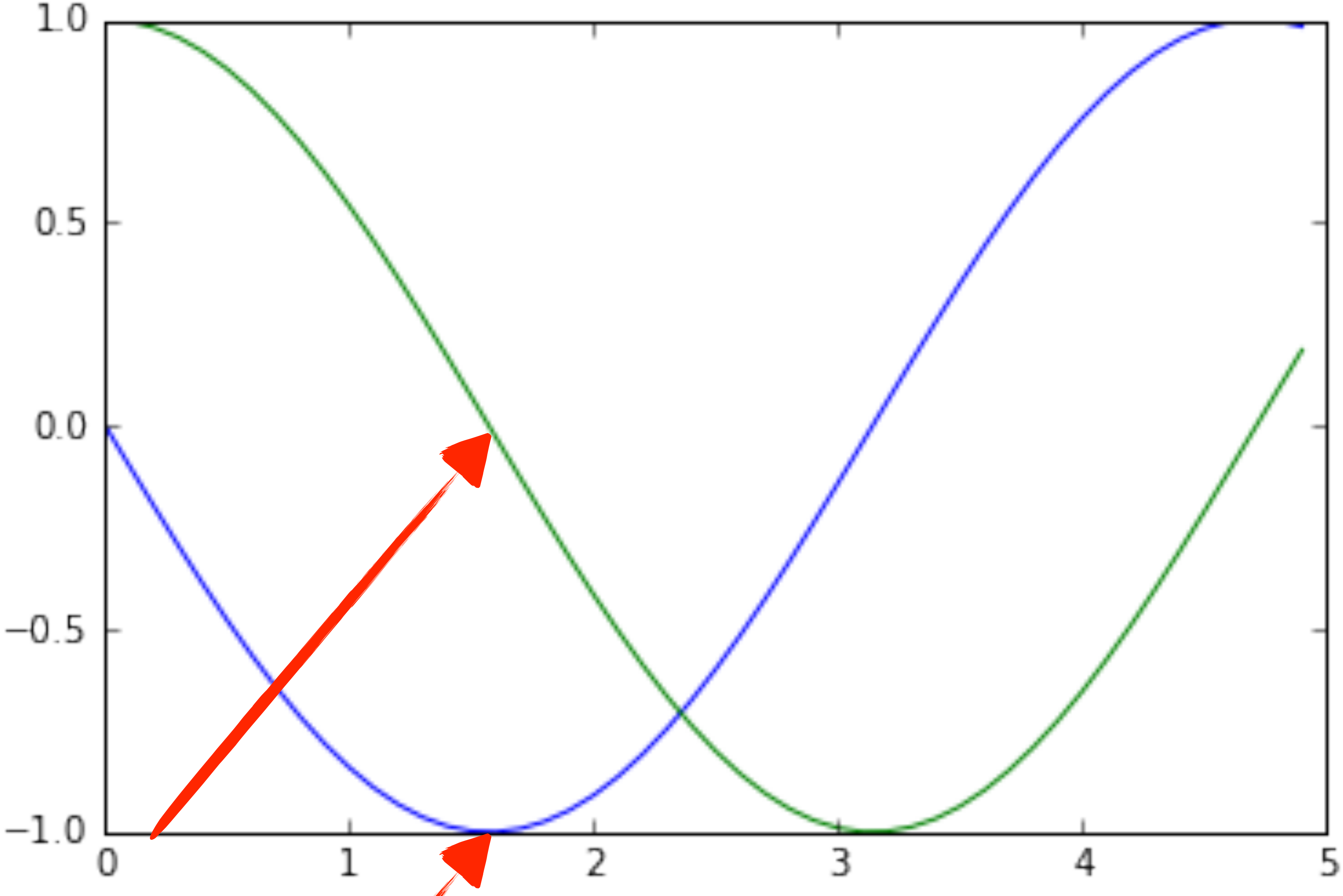
a3_m1_s2_v3_autodiff_v1

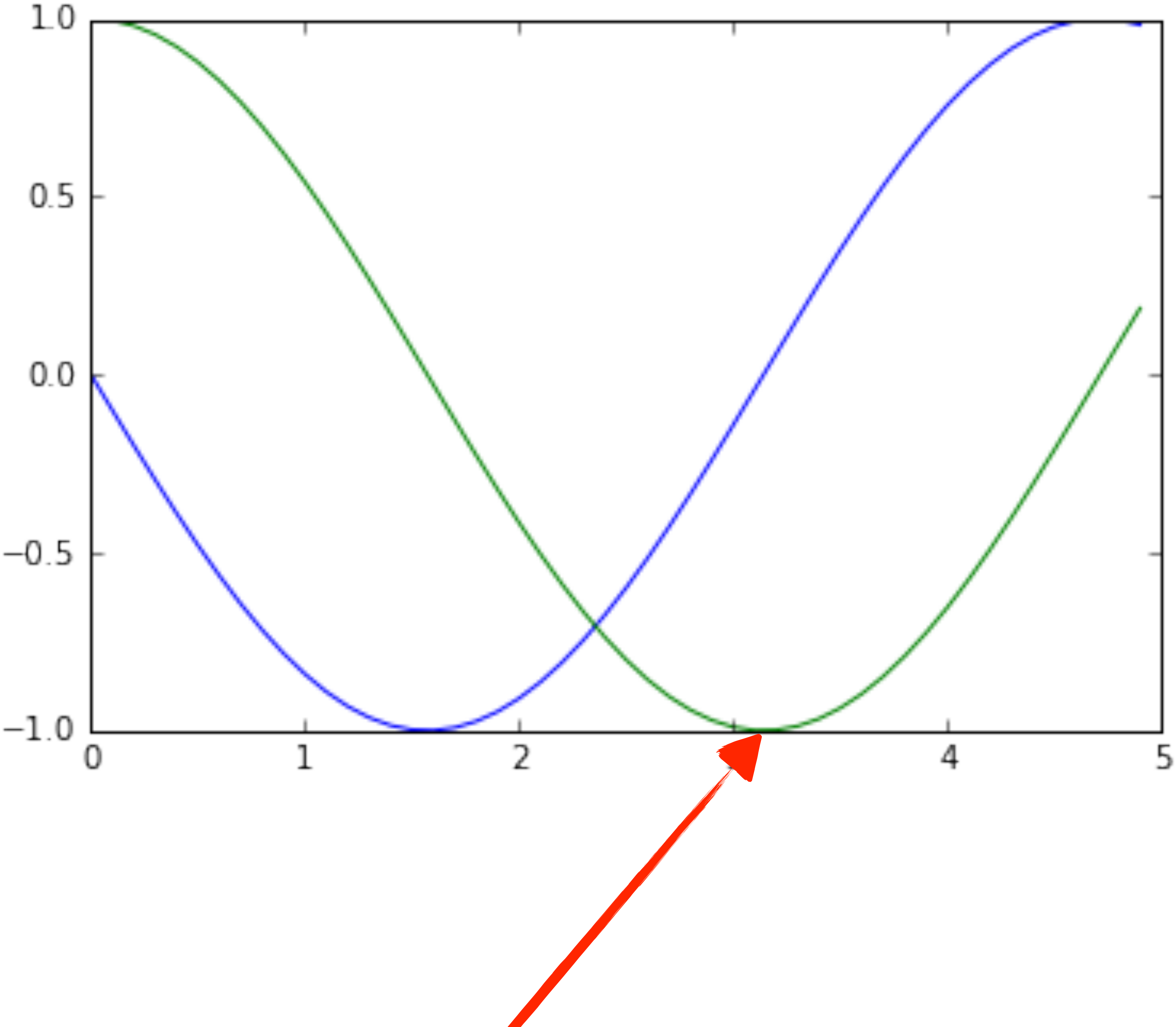

1.0
0.5
0.0
−0.5
−1.0
0
1
2
3
4
5

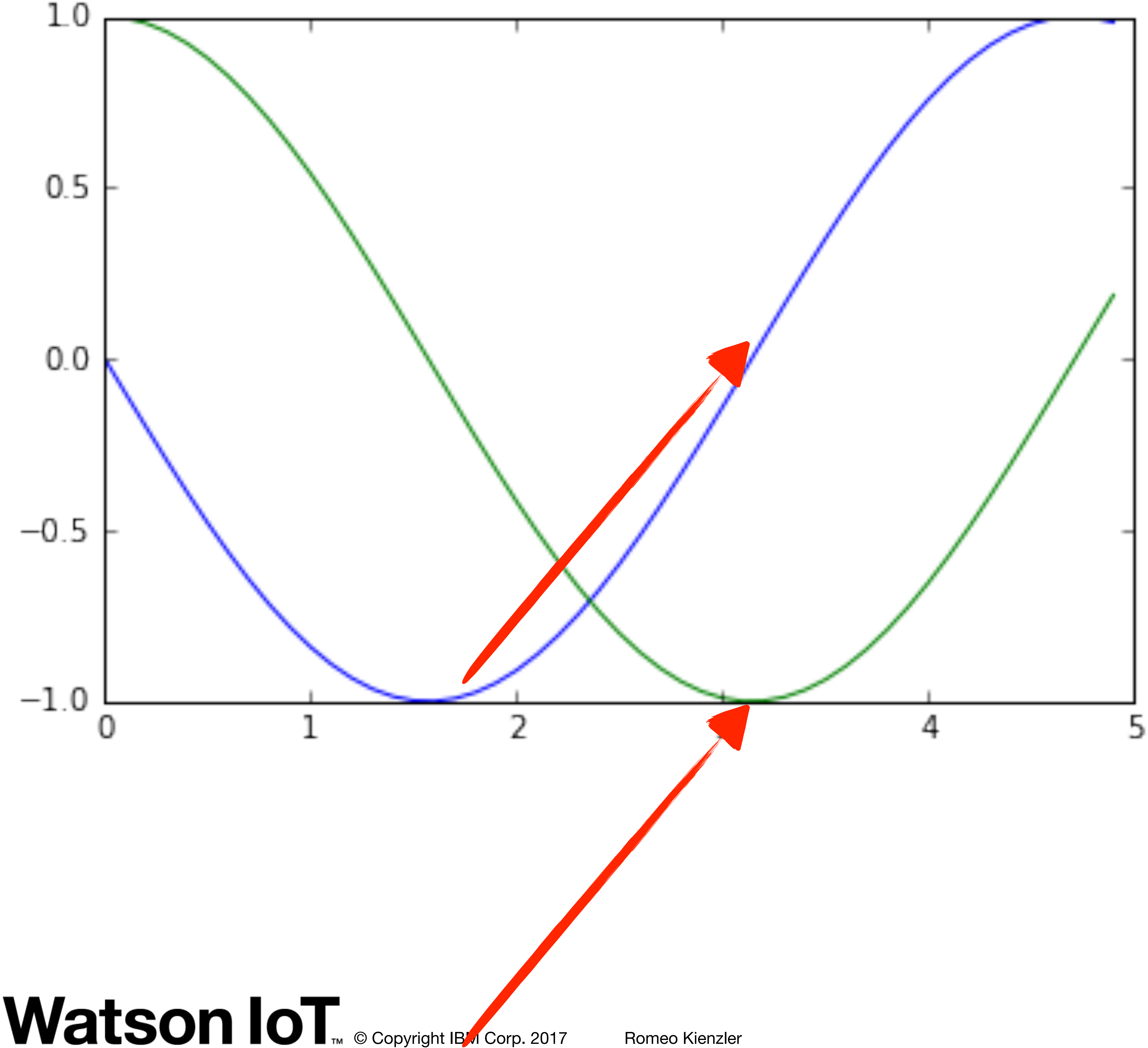

1.0
0.5
0.0
−0.5
−1.0
0
1
2
3
4
5

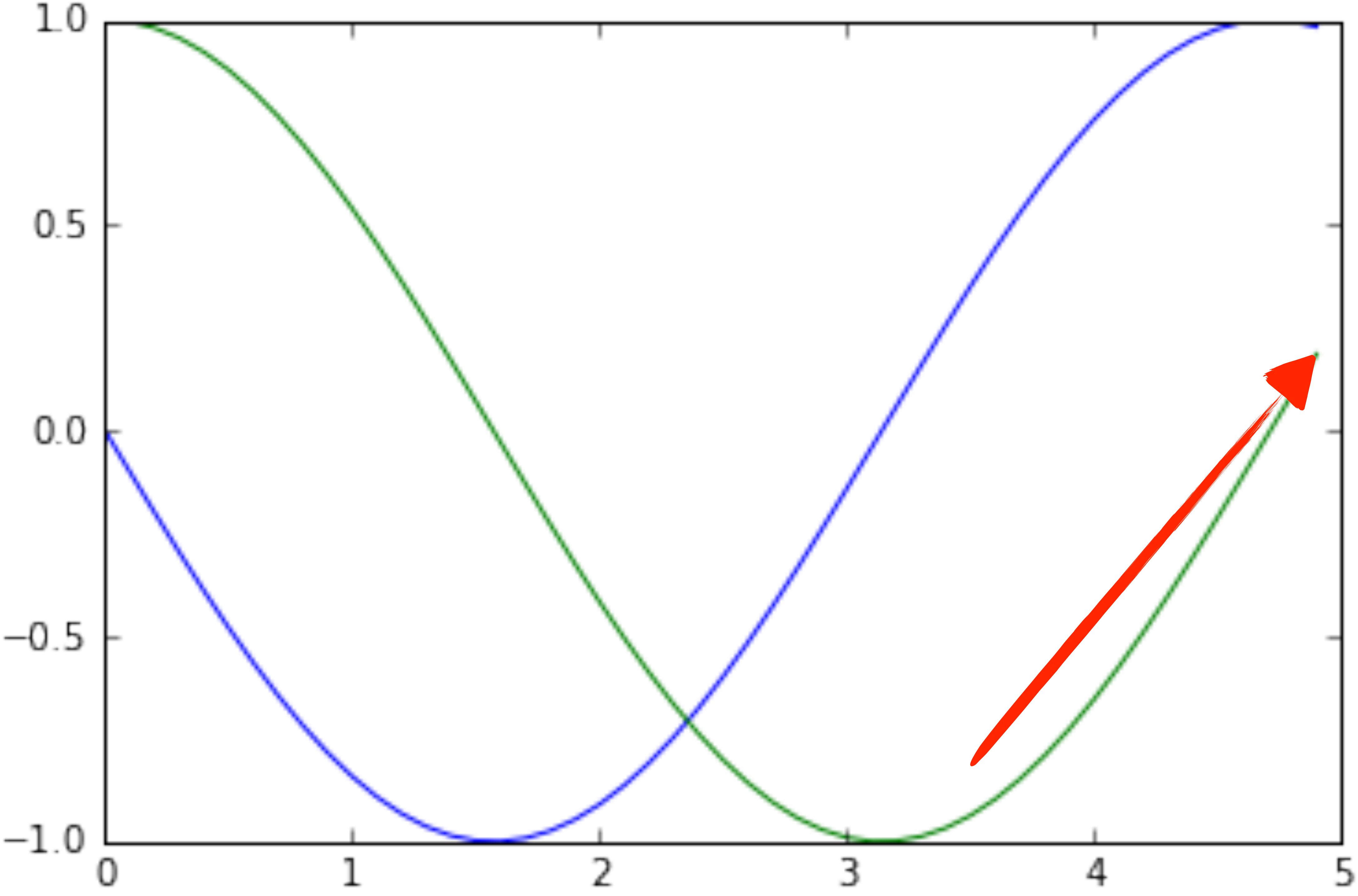

1.0
0.5
0.0
−0.5
−1.0
0
1
2
3
4
5

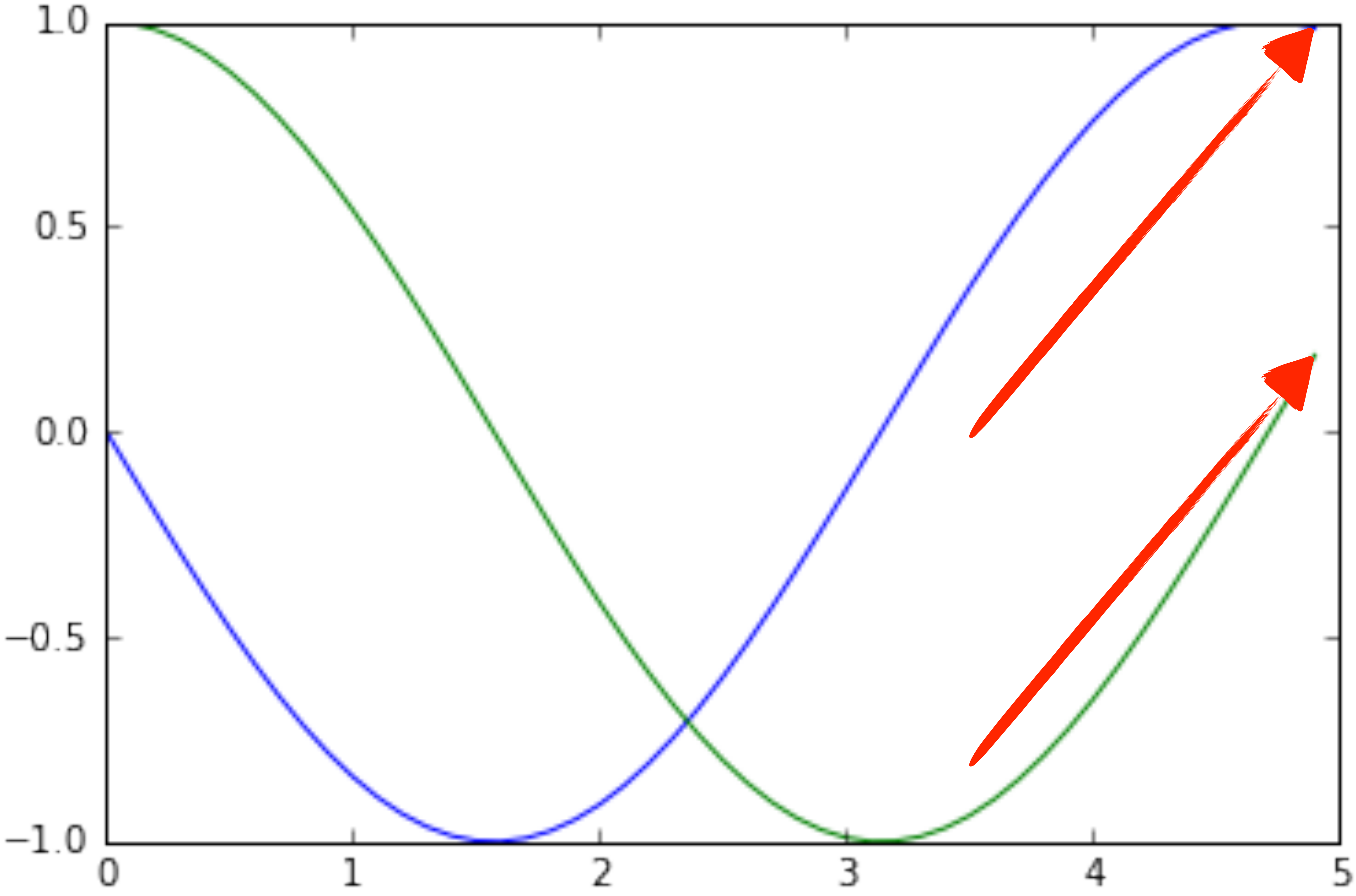

1.0
0.5
0.0
−0.5
−1.0
0
1
2
3
4
5


1.0
0.5
0.0
−0.5
−1.0
0
1
2
3
4
5


1.0
0.5
0.0
−0.5
−1.0
0
1
2
4
5

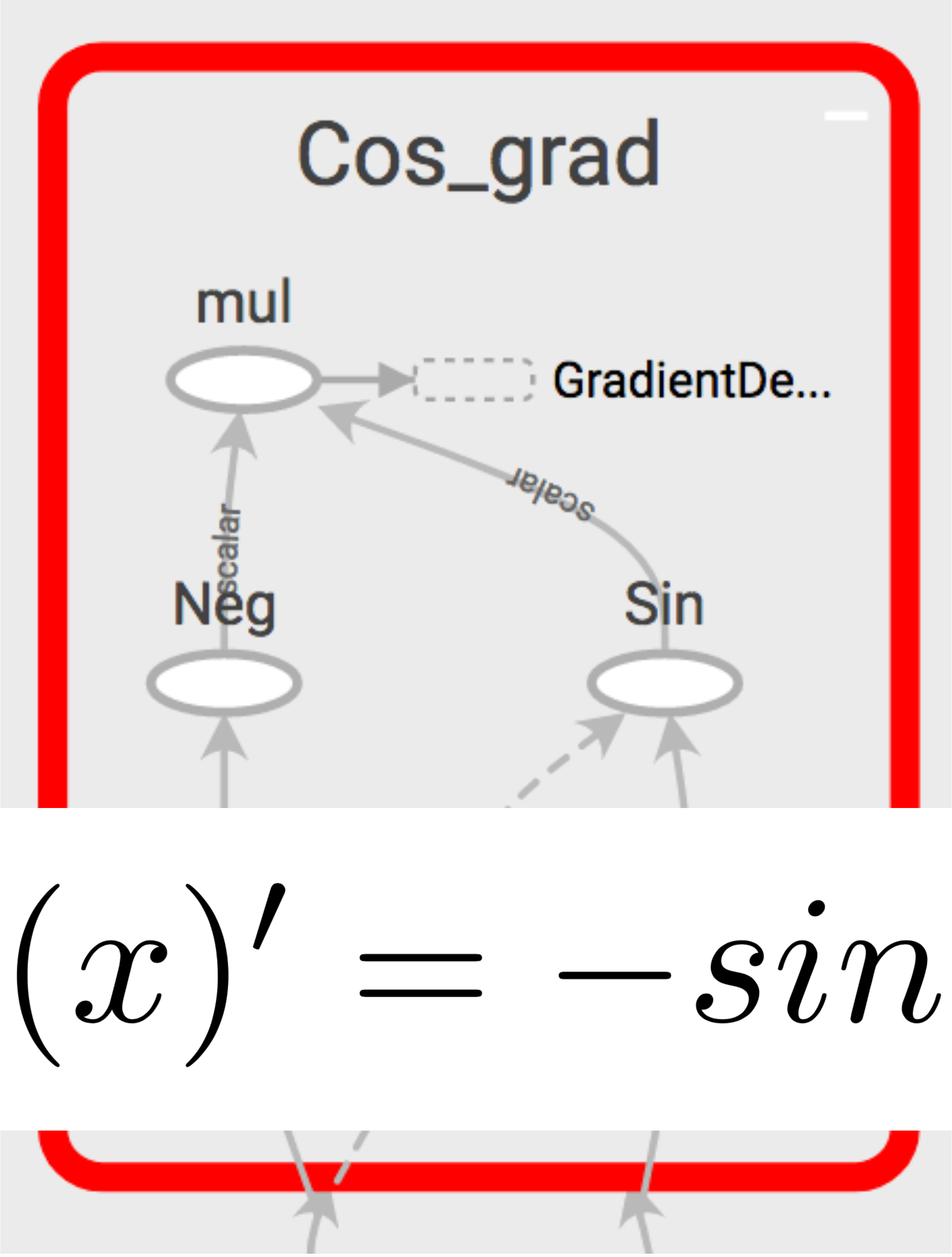

1.0
0.5
0.0
−0.5
−1.0
0
1
2
4
5


1.0
0.5
0.0
−0.5
−1.0
0
1
2
3
4
5


1.0
0.5
0.0
−0.5
−1.0
0
1
2
3
4
5

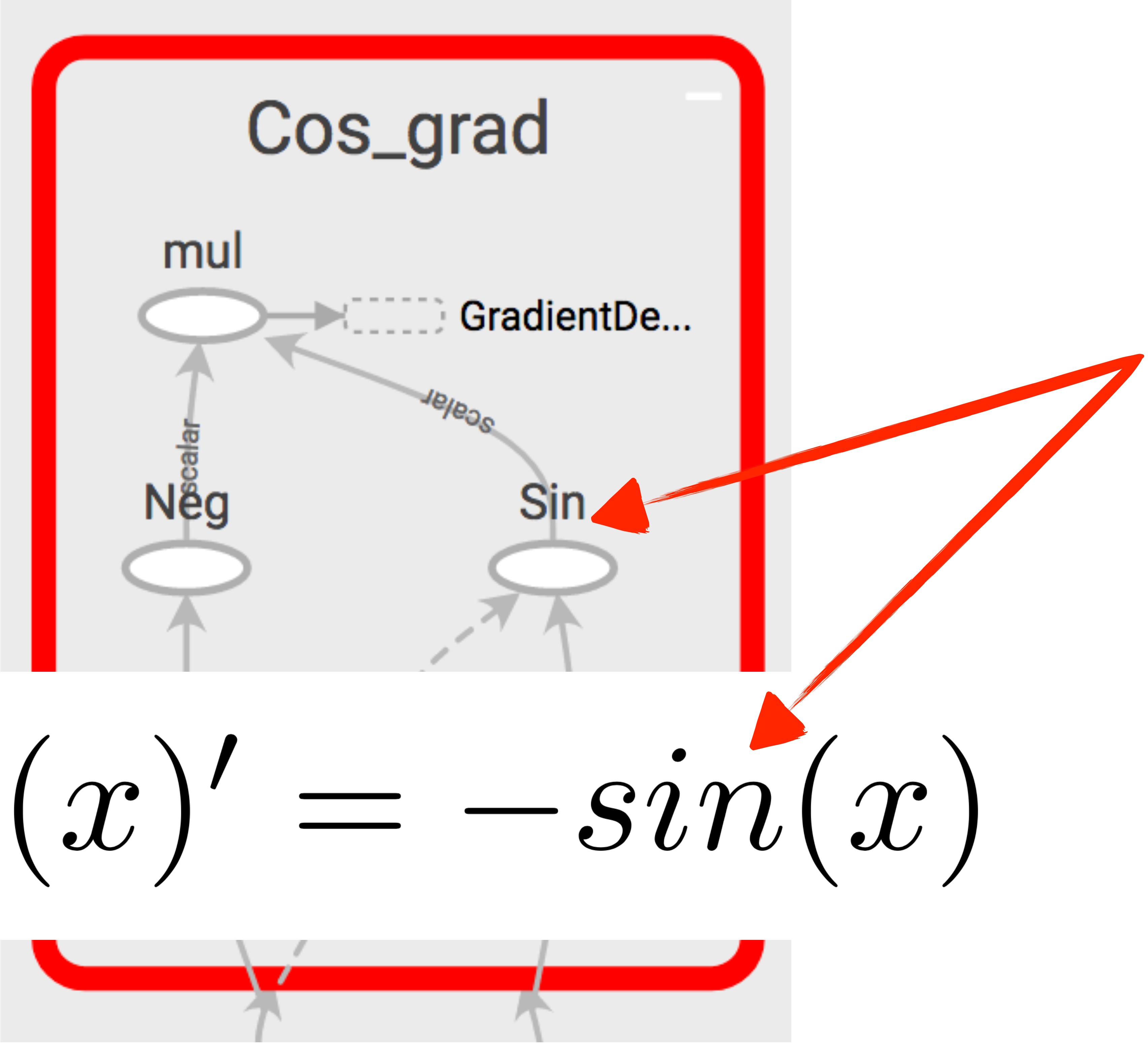
a3_m1_s2_v3_autodiff_v2

a3_m1_s2_v3_autodiff_v2

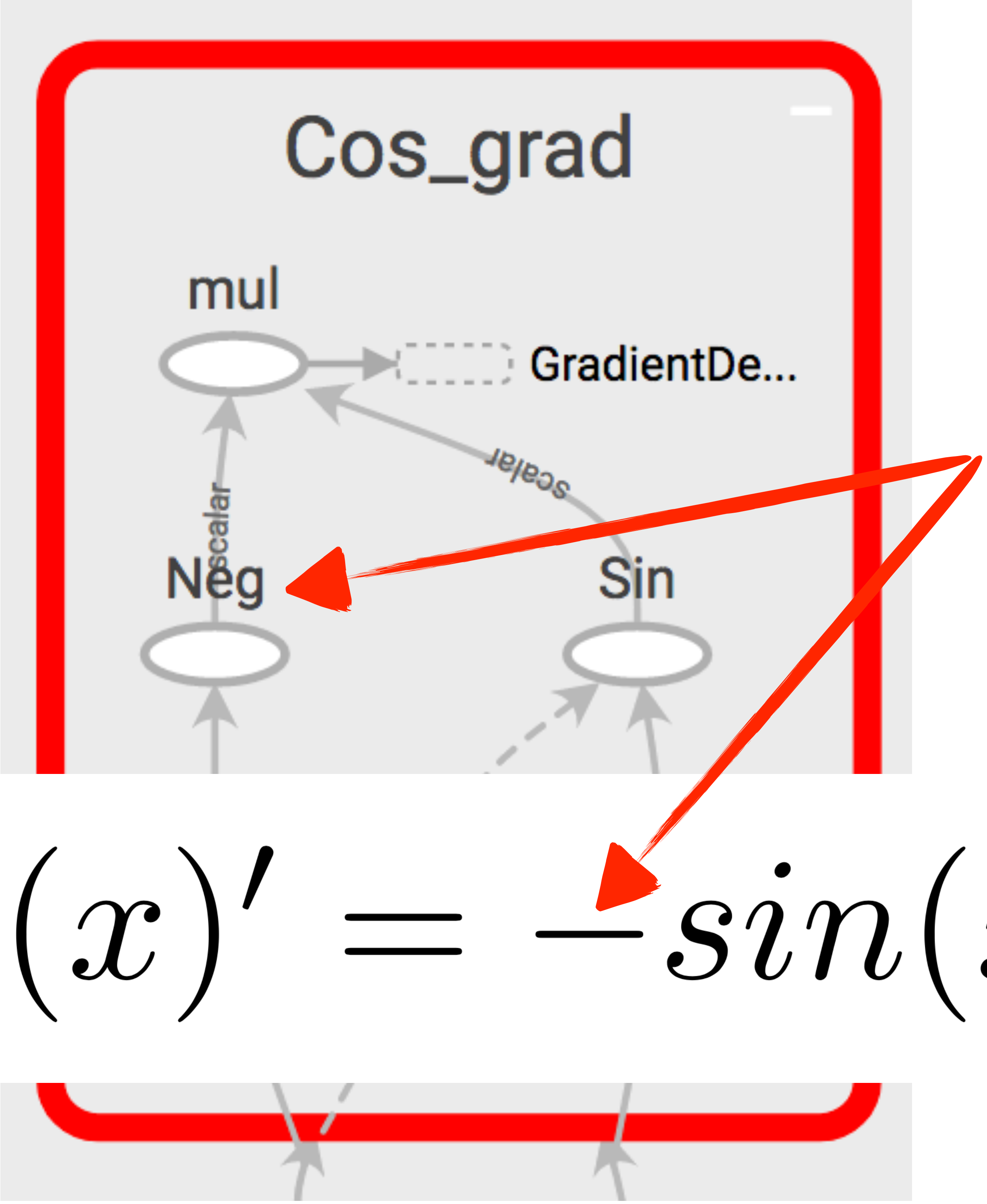


$$cos(x)' = -sin(x)$$

Credits to Salvador Dali for this example
https://stackoverflow.com/questions/44342432/is-gradient-in-the-tensorflows-graph-calculated-incorrectly



$$cos(x)' = -sin(x)$$

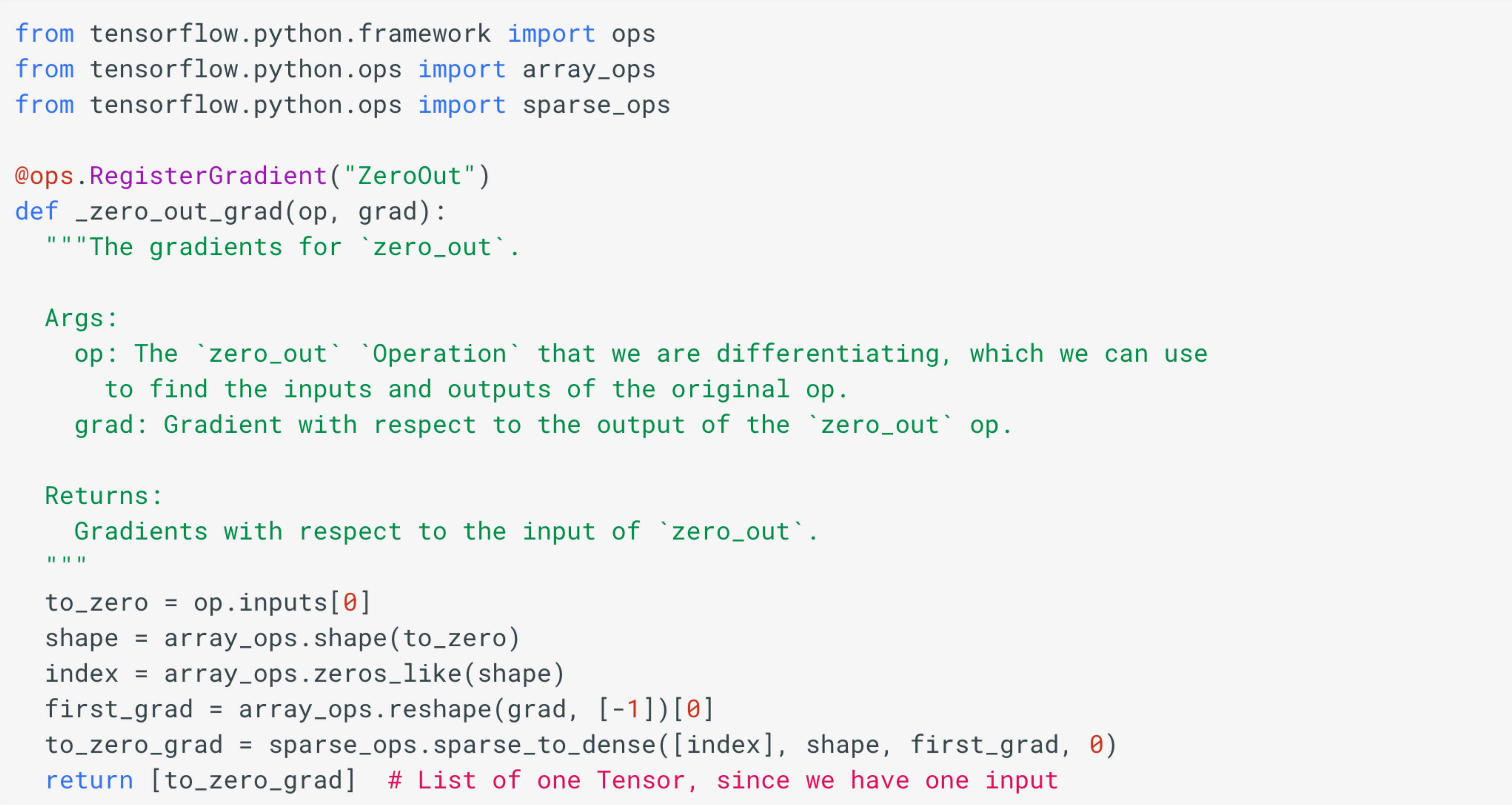


$$cos(x)' = -sin(x)$$

]

$$f(x)' = g(x)$$

$$\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx} = f'(y)g'(x) = f'(g(x))g'(x)$$



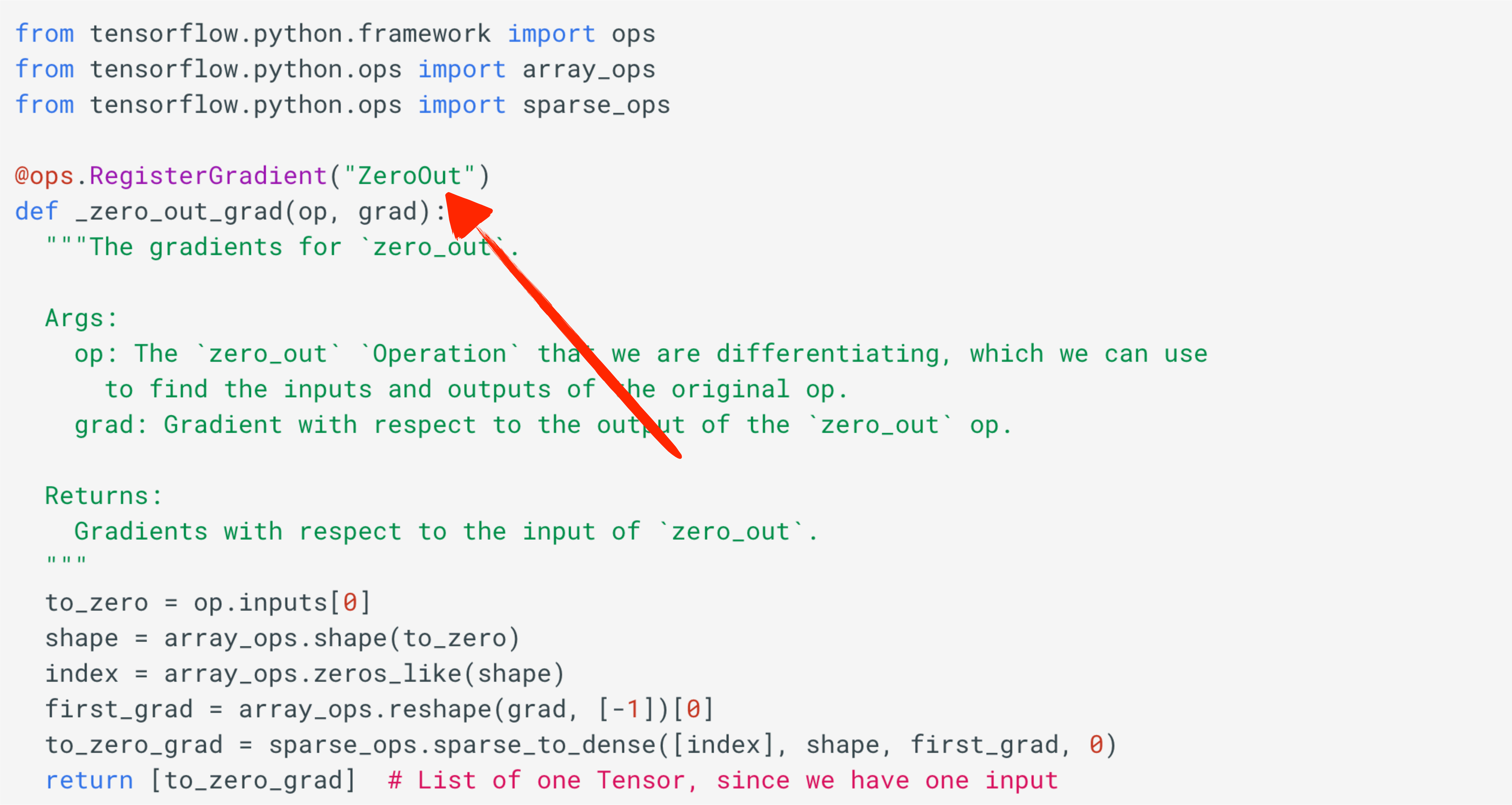
```python
from tensorflow.python.framework import ops
from tensorflow.python.ops import array_ops
from tensorflow.python.ops import sparse_ops

@ops.RegisterGradient("ZeroOut")
def _zero_out_grad(op, grad):
  """The gradients for `zero_out`.

  Args:
    op: The `zero_out` `Operation` that we are differentiating, which we can use
      to find the inputs and outputs of the original op.
    grad: Gradient with respect to the output of the `zero_out` op.

  Returns:
    Gradients with respect to the input of `zero_out`.
  """
  to_zero = op.inputs[0]
  shape = array_ops.shape(to_zero)
  index = array_ops.zeros_like(shape)
  first_grad = array_ops.reshape(grad, [-1])[0]
  to_zero_grad = sparse_ops.sparse_to_dense([index], shape, first_grad, 0)
  return [to_zero_grad]  # List of one Tensor, since we have one input
```

Credits: https://www.tensorflow.org/extend/adding_an_op



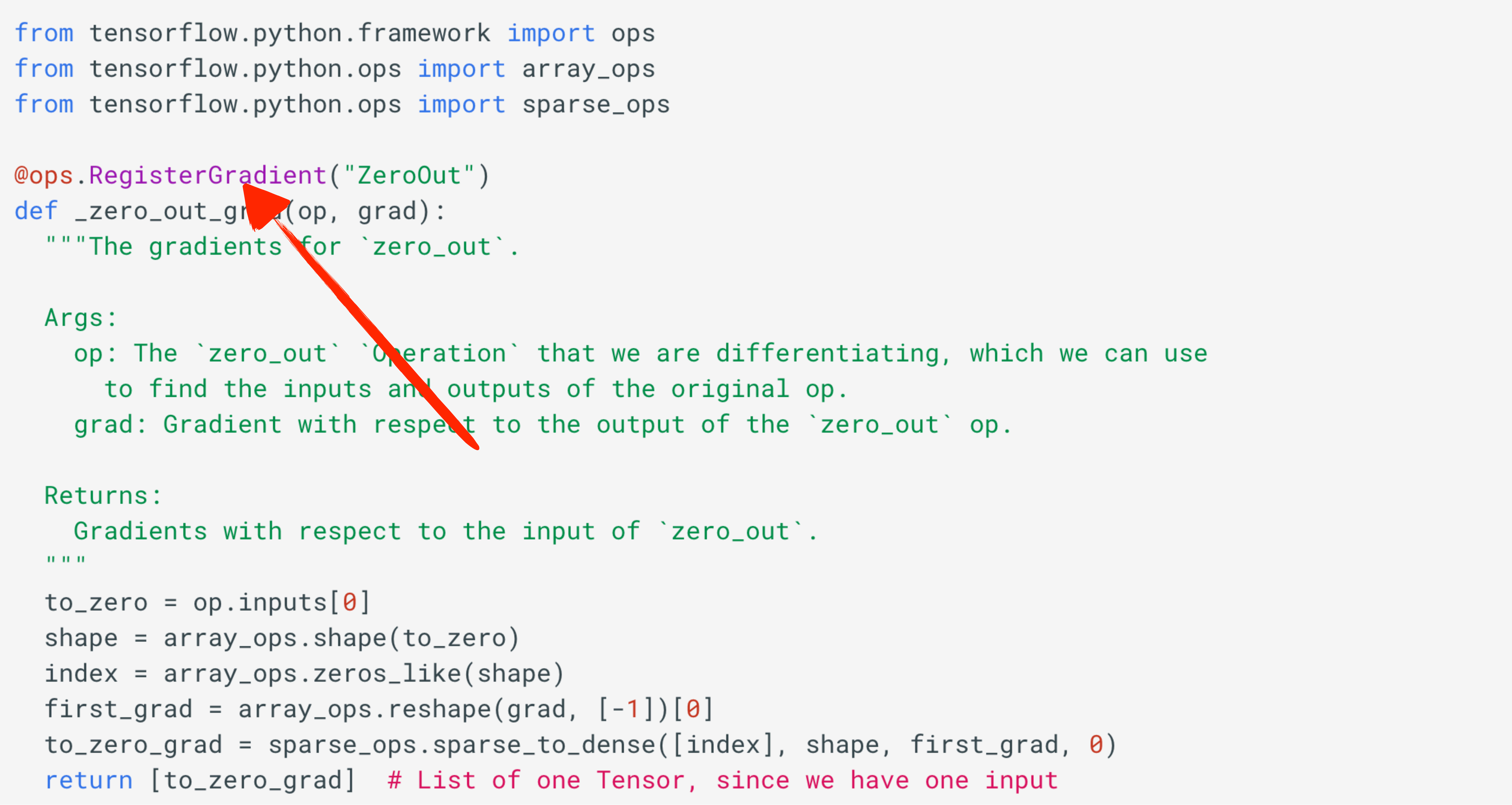
```python
from tensorflow.python.framework import ops
from tensorflow.python.ops import array_ops
from tensorflow.python.ops import sparse_ops

@ops.RegisterGradient("ZeroOut")
def _zero_out_grad(op, grad):
  """The gradients for `zero_out`.

  Args:
    op: The `zero_out` `Operation` that we are differentiating, which we can use
      to find the inputs and outputs of the original op.
    grad: Gradient with respect to the output of the `zero_out` op.

  Returns:
    Gradients with respect to the input of `zero_out`.
  """
  to_zero = op.inputs[0]
  shape = array_ops.shape(to_zero)
  index = array_ops.zeros_like(shape)
  first_grad = array_ops.reshape(grad, [-1])[0]
  to_zero_grad = sparse_ops.sparse_to_dense([index], shape, first_grad, 0)
  return [to_zero_grad]  # List of one Tensor, since we have one input
```

Credits: https://www.tensorflow.org/extend/adding_an_op



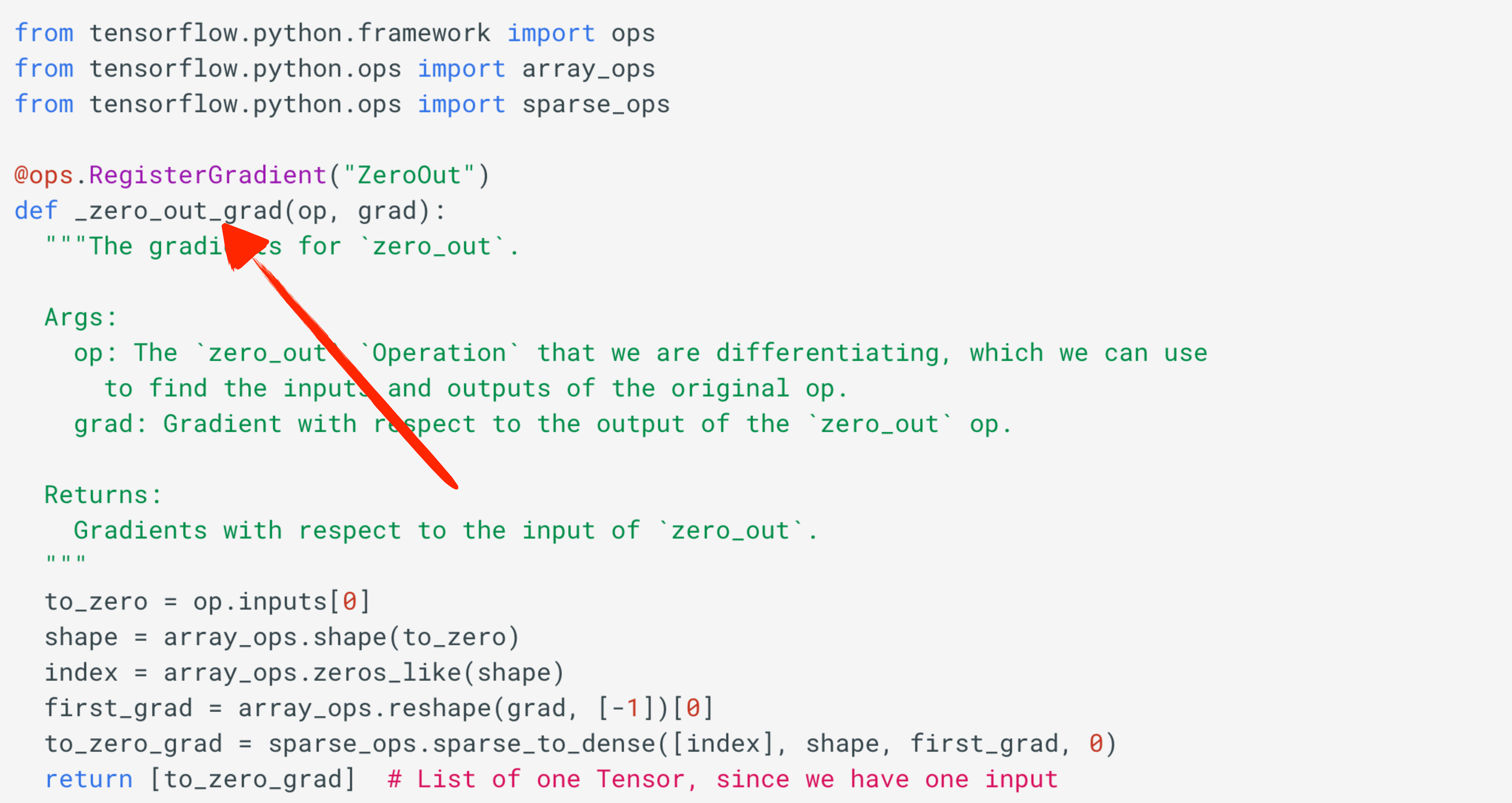
```python
from tensorflow.python.framework import ops
from tensorflow.python.ops import array_ops
from tensorflow.python.ops import sparse_ops

@ops.RegisterGradient("ZeroOut")
def _zero_out_grad(op, grad):
  """The gradients for `zero_out`.

  Args:
    op: The `zero_out` `Operation` that we are differentiating, which we can use
      to find the inputs and outputs of the original op.
    grad: Gradient with respect to the output of the `zero_out` op.

  Returns:
    Gradients with respect to the input of `zero_out`.
  """
  to_zero = op.inputs[0]
  shape = array_ops.shape(to_zero)
  index = array_ops.zeros_like(shape)
  first_grad = array_ops.reshape(grad, [-1])[0]
  to_zero_grad = sparse_ops.sparse_to_dense([index], shape, first_grad, 0)
  return [to_zero_grad]  # List of one Tensor, since we have one input
```

Credits: https://www.tensorflow.org/extend/adding_an_op



```python
from tensorflow.python.framework import ops
from tensorflow.python.ops import array_ops
from tensorflow.python.ops import sparse_ops

@ops.RegisterGradient("ZeroOut")
def _zero_out_grad(op, grad):
  """The gradients for `zero_out`.

  Args:
    op: The `zero_out` `Operation` that we are differentiating, which we can use
      to find the inputs and outputs of the original op.
    grad: Gradient with respect to the output of the `zero_out` op.

  Returns:
    Gradients with respect to the input of `zero_out`.
  """
  to_zero = op.inputs[0]
  shape = array_ops.shape(to_zero)
  index = array_ops.zeros_like(shape)
  first_grad = array_ops.reshape(grad, [-1])[0]
  to_zero_grad = sparse_ops.sparse_to_dense([index], shape, first_grad, 0)
  return [to_zero_grad]  # List of one Tensor, since we have one input
```

Credits: https://www.tensorflow.org/extend/adding_an_op

Summary