```
import numpy as np
import numpy.random as npr
from test_util import *
from funkyyak import grad
npr.seed(1)

def test_abs():
    fun = lambda x : 3.0 * np.abs(x)
    d_fun = grad(fun)
    check_grads(fun, 1.1)
    check_grads(fun, -1.1)
    check_grads(d_fun, 1.1)
    check_grads(d_fun, -1.1)

def test_sin():
    fun = lambda x : 3.0 * np.sin(x)
    d_fun = grad(fun)
    check_grads(fun, npr.randn())
    check_grads(d_fun, npr.randn())

def test_sign():
    fun = lambda x : 3.0 * np.sign(x)
    d_fun = grad(fun)
    check_grads(fun, 1.1)
    check_grads(fun, -1.1)
    check_grads(d_fun, 1.1)
    check_grads(d_fun, -1.1)

def test_exp():
    fun = lambda x : 3.0 * np.exp(x)
    d_fun = grad(fun)
    check_grads(fun, npr.randn())
    check_grads(d_fun, npr.randn())
```

```
34
35  def test_log():
36      fun = lambda x : 3.0 * np.log(x)
37      d_fun = grad(fun)
38      check_grads(fun, abs(npr.randn()))
39      check_grads(d_fun, abs(npr.randn()))
40
41  def test_neg():
42      fun = lambda x : 3.0 * - x
43      d_fun = grad(fun)
44      check_grads(fun, npr.randn())
45      check_grads(d_fun, npr.randn())
46
47  def test_cos():
48      fun = lambda x : 3.0 * np.cos(x)
49      d_fun = grad(fun)
50      check_grads(fun, npr.randn())
51      check_grads(d_fun, npr.randn())
52
53  def test_tan():
54      fun = lambda x : 3.0 * np.tan(x)
55      d_fun = grad(fun)
56      check_grads(fun, npr.randn())
57      check_grads(d_fun, npr.randn())
58
59  def test_cosh():
60      fun = lambda x : 3.0 * np.cosh(x)
61      d_fun = grad(fun)
62      check_grads(fun, npr.randn())
63      check_grads(d_fun, npr.randn())
64
65  def test_sinh():
66      fun = lambda x : 3.0 * np.sinh(x)
67      d_fun = grad(fun)
68      check_grads(fun, npr.randn())
69      check_grads(d_fun, npr.randn())
70
71  def test_tanh():
72      fun = lambda x : 3.0 * np.tanh(x)
73      d_fun = grad(fun)
74      check_grads(fun, npr.randn())
75      check_grads(d_fun, npr.randn())
```